(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平8－314872

(43)公開日　平成８年(1996)11月29日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ	15/16	３７０		Ｇ０６Ｆ 15/16	３７０Ｚ	
	9/46	３６０		9/46	３６０Ｆ	
	15/00	３９０	9364－5L	15/00	３９０	

審査請求　有　　請求項の数７　ＦＤ　（全 32 頁）

(21)出願番号　特願平7－138641

(22)出願日　平成７年(1995) ５月12日

(71)出願人　000233538
日立東北ソフトウェア株式会社
宮城県仙台市青葉区一番町２丁目４番１号
(72)発明者　伊藤　俊明
宮城県仙台市青葉区一番町二丁目４番１号
日立東北ソフトウェア株式会社内
(72)発明者　樋地　正浩
宮城県仙台市青葉区一番町二丁目４番１号
日立東北ソフトウェア株式会社内
(72)発明者　岡崎　司
宮城県仙台市青葉区一番町二丁目４番１号
日立東北ソフトウェア株式会社内
(74)代理人　弁理士　須田　篤

最終頁に続く

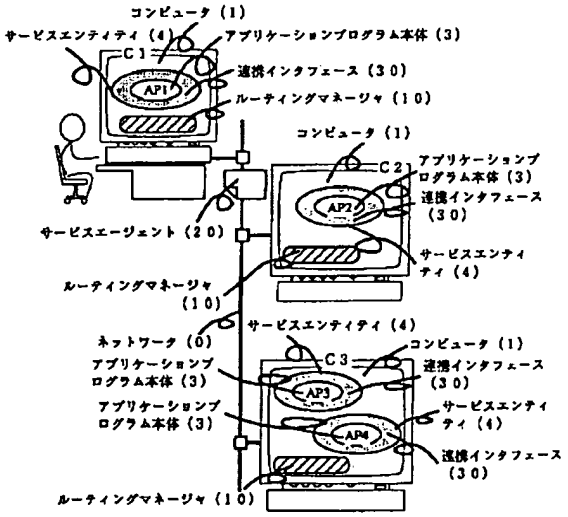
(54)【発明の名称】　アプリケーションプログラム間連携処理方法

(57)【要約】

【目的】個々の利用者が自分自身の利用方法に適するアプリケーションプログラム間の連携処理を行うことを可能とする。

【構成】複数のアプリケーションプログラム３の利用手順とその各々の利用手続きを記述した作業指示書（サービスエージェント）２０を作成し、該作業指示書を複数のコンピュータ１へ順次転送する。各アプリケーションプログラム３には、作業指示書に記載された利用手続きに応じて当該アプリケーションプログラムとの連携をとる連携インタフェース３０を設けておき、各連携インタフェース３０は、作業指示書２０を受けたとき、該連携インタフェースが当該作業指示書に記載された自己のアプリケーションプログラムに関する利用手続きに応じて自己のアプリケーションプログラムを起動し、目的とする制御および／またはデータの授受を行う。

図１　アプリケーションプログラム間連携処理システム構成



【特許請求の範囲】
【請求項１】コンピュータネットワークに接続された複数のコンピュータ上で動作する複数のアプリケーションプログラム間の連携処理方法において、
前記複数のアプリケーションプログラムの利用順序と各アプリケーションプログラムの利用手続きを記述した作業指示書を作成し、
該作業指示書を、前記コンピュータネットワークを介して、目的のアプリケーションプログラムを有する複数のコンピュータへ順次転送し、
前記複数のアプリケーションプログラムの各々には、前記作業指示書に記載された当該アプリケーションプログラムの利用手続きに応じて当該アプリケーションプログラムとの連携をとる連携インタフェースを設けておき、
各連携インタフェースは、前記作業指示書を受けたとき、該連携インタフェースが当該作業指示書に記載された自己のアプリケーションプログラムに関する利用手続きに応じて自己のアプリケーションプログラムを起動し、目的とする制御および／またはデータの授受を行い、
必要なアプリケーションプログラムを有するコンピュータを作業指示書が一巡することにより前記複数のアプリケーションプログラムによる一連の作業を実行することを特徴とするアプリケーションプログラム間連携処理方法。
【請求項２】請求項１記載のアプリケーションプログラム間連携処理方法において、前記作業指示書に記載された利用手続きの構成要素として、当該作業指示書が利用する全てのアプリケーションプログラムで使用されるデータ項目の一覧を表す全データ項目、各アプリケーションプログラムへ与えるべきデータが格納されている出力データ項目、各アプリケーションプログラムの処理結果が格納される入力データ項目、前記出力データ項目に格納されているデータを当該アプリケーションプログラムへ出力する手続きを表すデータ出力手続き、前記入力データ項目ごとのデータの入力手続きを表すデータ入力手続き、該データ入力手続きを用いて入力されたデータを処理する入力データ処理手続き、当該アプリケーションプログラムの実行を制御する制御手続き、のうち少なくとも１つを含むことを特徴とするアプリケーションプログラム間連携処理方法。
【請求項３】請求項１または２記載のアプリケーションプログラム間連携処理方法において、各コンピュータ上に移動制御手段を有し、該移動制御手段が、自己のコンピュータ上の前記連携インタフェース及び他のコンピュータとの間の通信路を確保、管理し、該通信路を通して受信した前記作業指示書に記載された利用順序に従って当該作業指示書の移動を制御することを特徴とするアプリケーションプログラム間連携処理方法。
【請求項４】請求項２記載のアプリケーションプログラム間連携処理方法において、前記作業指示書の構成要素である制御手続きとして、該作業指示書を受信した前記連携インタフェースのアプリケーションプログラムを起動する起動手続き、該アプリケーションプログラムの実行を終了する終了手続き、該アプリケーションプログラムの有する１つ以上のコマンドを実行する実行手続き、それらを組み合せた制御スクリプトの少なくとも１つの手続きを有することを特徴とするアプリケーションプログラム間連携処理方法。
【請求項５】請求項２記載のアプリケーションプログラム間連携処理方法において、前記作業指示書の構成要素として、新たな作業指示書を生成するための生成手続きを有することを特徴とするアプリケーションプログラム間連携処理方法。
【請求項６】請求項１または２記載のアプリケーションプログラム間連携処理方法において、前記利用手続きは、プログラムおよび関数名のいずれかであり、関数名の場合には前記連携インタフェースにおいて当該関数名に対応するプログラムを取得することを特徴とするアプリケーションプログラム間連携処理方法。
【請求項７】請求項１，２，３，４，５または６記載のアプリケーションプログラム間連携処理方法において、前記作業指示書に記述された利用順序は、コンピュータ名と、そのコンピュータ上で動作するアプリケーションプログラムおよび連携インタフェースからなるサービスエンティティの名称との組を該サービスエンティティの利用の順序に従って指定したものであり、前記コンピュータ名を省略可能としたことを特徴とするアプリケーションプログラム間連携処理方法。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、利用者が複数のアプリケーションプログラムを順次組み合わせながら利用して処理を行う分散処理システムに係り、必要に応じてその処理経過や処理結果、処理内容を他の作業を行っている各作業者との間でやり取りすることにより、お互いの作業を連携させながら進めていく一連の作業から構成される業務において、これらの業務を行う際に各部署や作業者が利用するさまざまなアプリケーションプログラムの間の処理を連携させ、さらに業務に応じて個々のアプリケーションプログラムの間の処理の連携手続きを容易に変更することを可能とするアプリケーションプログラム間の連携処理方法に関する。
【０００２】特に、コンピュータネットワークに接続されたコンピュータ上で動作する各種アプリケーションプログラムにより構成される分散処理システムにおいて、利用者が行っていた複数のアプリケーションプログラムを組み合わせた操作手順や、各作業者に依頼して行っていたアプリケーションプログラムの操作やその結果の取得を自動化するとともに、一連の作業に必要なアプリケ

ーションプログラムの利用手順をシステム化することにより、各作業者の作業負荷を軽減し、作業の流れを円滑にするために、作業に応じて柔軟にアプリケーションプログラムを組み合わせ、連携して処理するための利用技術、及びこのような柔軟なアプリケーションプログラムの連携処理方法を実現するための分散処理システムに関する。

【０００３】

【従来の技術】分散処理システムは、個々の作業者が行うべき作業に用いられるコンピュータをネットワークで接続し、個々の作業者間で必要とするデータをやり取りするシステムである。このようなシステムでは、１つ１つの作業は個々の作業者がアプリケーションプログラムを用いて行う個人利用中心であり、各作業者の間で必要とするデータだけが共有され、各作業者の利用するアプリケーションプログラムで共有されたデータを利用できる。

【０００４】分散処理システム上でアプリケーションプログラムを連携して動作させるための仕組みに関しては、日経コンピュータ：「クライアント・サーバーの弱点 非同期、蓄積型通信が解消」1994.8.8. pp.159-168、日経コンピュータ：「分散アプリケーション同士の非同期連携を容易に実現する　メッセージ・ベースのミドルウェアが急増」1993.9.20. pp.67-74 に、RPC（Remote Procedure Call）、会話型、ファイル転送、メッセージ・ベースの４つの連携処理方式について論じられている。

【０００５】「RPC」は、アプリケーションプログラムの１つのモジュールを取りだし、そのモジュールを他のコンピュータで実行させる方式である。「会話型」は、複数のアプリケーションプログラムの間をリアルタイム通信路で接続し、そのリアルタイム通信路を通じてデータや処理要求をやり取りする方式である。会話型の方式を用いて、或る作業に利用するアプリケーションプログラムを処理の要求元（クライアント）と処理の実行元（サーバ）に分割しクライアント側アプリケーションプログラムとサーバ側アプリケーションプログラムの間で連携して処理を行えるようにしたものがクライアント・サーバ・システムである。これらの２つの方式では、受信側アプリケーションプログラムは常に動作し、送信側アプリケーションプログラムからの処理要求を待つ状態にあり、送信側アプリケーションプログラムからの処理要求を受け取るとそれを即座に処理し、処理結果を送信側アプリケーションプログラムに返すことにより、送信側アプリケーションプログラムと受信側アプリケーションプログラムの間で同期を取りながら処理を行う連携処理を実現できる。また、これらの方式では、連携処理を行うアプリケーションプログラムの間の連携手続きをアプリケーションプログラムを開発する際にあらかじめ定め、その手続きをプログラムとして記述しておく必要がある。

【０００６】「ファイル転送」は、連携させるアプリケーションプログラムの間でファイルを転送し、ファイルに記述されたデータを受け取った場合にアプリケーションプログラムがそのデータに対する処理を行う方式である。この方式では、ファイルの受信をトリガとしてアプリケーションプログラムを実行させることにより、非同期的なアプリケーションプログラム間の連携処理を実現できる。「メッセージ・ベース」は、送信側アプリケーションプログラムから送り出されたメッセージをメッセージ・ベースの処理を実現するプログラムが受け取り、メッセージに記述された受信側アプリケーションプログラムにそのメッセージを送り、メッセージを受け取った受信側アプリケーションプログラムではメッセージに対応した処理手続きを実行することにより、アプリケーションプログラム間の連携処理を実現する方式である。メッセージ・ベースの方式では、連携処理を行うアプリケーションプログラムの間であらかじめメッセージの形式を定め、そのメッセージを受け取った場合のアプリケーションプログラムの処理を記述する。

【０００７】これら４つのアプリケーションプログラム連携処理方式では、一連の作業から構成される業務を、個々の作業で用いられる複数のアプリケーションプログラムの間で処理を連携させることにより行うためには、連携のために必要な処理手続きを各アプリケーションプログラムごとに定め、プログラムとして記述しておく必要がある。

【０００８】特開平2-186737号公報に記載のプログラム間論理通信路制御方式では、ネットワークに接続されたコンピュータ上で動作するアプリケーションプログラムの間で、データを交換することにより、アプリケーションプログラム間の連携処理を実現する方式について述べられている。この方式では、アプリケーションプログラムの画面を介してユーザにより加えられた操作（データの変更、追加、削除など）のデータを論理的通信路を通じて同一のアプリケーションプログラム間でやり取りすることにより、そのアプリケーションプログラム間で画面の表示を共有することに基づくアプリケーションプログラム間の連携処理を実現している。

【０００９】これらのアプリケーションプログラム間の連携処理方式においては、連携処理に用いるデータやメッセージを他のコンピュータ上のアプリケーションプログラムに送る際には、送信先のコンピュータの名称、送信先コンピュータの物理的な位置または物理的名称のいずれか１つの方法でデータやメッセージの送信先を指定することにより、指定されたコンピュータ上のアプリケーションプログラムにデータやメッセージを送ることができる。

【００１０】最近では、個々の作業者による個人利用中心のシステムから複数の作業者の間にまたがる作業を支

援するシステムの研究・開発が盛んに行われている。これら複数の作業者に跨る作業を支援するシステムの種類とその支援する作業の内容に関しては、石井裕：「グループウェア技術の研究動向」情報処理学会論文誌 Vol.30 No.12 Dec.1989 、石井裕：「コンピュータを用いたグループワーク支援の研究動向」コンピュータソフトウェア Vol.8 No.2 pp.14-26、C.A. Ellis , S.J. Gibbs and G.L.Rein : " Groupware : Some Issues and Experiences",CACM Jan.1991 Vol.34 , No.1 pp.38-58.、に国内外で研究、開発が行われている複数の人々の間に跨って行われる協同作業を支援するシステムが論じられている。

【００１１】これらの協同作業を支援するシステムは、各作業者が同一のアプリケーションプログラムを利用して同一の内容の作業を協同して行うこと（協同文書作成など）を支援するシステムであり、各作業者の間でアプリケーションプログラムの処理結果を表示し、処理結果に対して何らかの操作を加えるユーザインタフェース画面を共有する機能を提供する。

【００１２】

【発明が解決しようとする課題】上記従来技術は、或る特定のアプリケーションプログラム間でやり取りするデータやメッセージをあらかじめ定め、その定めたデータやメッセージに対応する処理プログラムをそれぞれのアプリケーションプログラムに記述し、このあらかじめ定められたデータやメッセージをアプリケーションプログラム間でやり取りすることにより、アプリケーションプログラム間の連携処理を行う。そのため、アプリケーションプログラム間の連携処理の手続きを変更することはできない。あるいは、変更できたとしても、変更に関連する全てのアプリケーションプログラム中の処理プログラムを変更する必要がある。その結果、作業内容や目的に応じて容易にアプリケーションプログラム間の連携処理手続きを変更することは非常に困難となっている。

【００１３】さらにアプリケーションプログラム間の連携処理手続きを変更するためには、その変更対象となる全てのアプリケーションプログラムの実行をいったん終了させる必要がある。分散処理システム上ではさまざまな利用者が各々の作業を行うためにアプリケーションプログラムを使用しており、変更対象となる全てのアプリケーションプログラムの実行をいったん終了させることは、変更対象となるアプリケーションプログラムの数が増加するに従い、困難さを増してくることになる。

【００１４】また、アプリケーションプログラム間で連携処理を行うためにやり取りされるものは、データもしくはメッセージであるため、それらデータやメッセージを受け取った際の処理手続きは、それを受け取った受信側アプリケーションプログラムにより決められており、送信側アプリケーションプログラムが受信側アプリケーションプログラムの処理手続きを変更することは不可能であった。また、受信側アプリケーションプログラムの提供する複数の機能を組み合わせた一連の処理手続き、すなわち受信側アプリケーションプログラムで一度に実行したい複数のコマンドの組み合わせを実行すること、を実現するためには、毎回、同期を取りながら必要な処理手続きを実行するためのメッセージをアプリケーションプログラムに送り、該当する手続きを実行し、その実行結果を送信側アプリケーションプログラムで組み合わせるか、あらかじめ受信側アプリケーションプログラムに複数のコマンドを実行するための特別なメッセージをプログラムとして記述しておく必要があり、開発効率が低下する。

【００１５】このようにアプリケーションプログラム間の連携処理手続きをあらかじめ記述しておく必要があるため、個々の利用者が自分自身の利用方法に適するアプリケーションプログラム間の連携処理手続きを記述することは困難であり、システムにより提供されるアプリケーションプログラム間の連携処理手続きを利用する以外には、アプリケーションプログラム間を連携して処理させることは困難であった。

【００１６】さらにこれらのデータやメッセージをアプリケーションプログラムに送るためには、そのアプリケーションプログラムの動作しているコンピュータをコンピュータの名称、またはIP アドレス等のコンピュータを一意に識別するための手段で明示的に指定する必要があり、或る処理を行うアプリケーションプログラムの動作するコンピュータを何等かの理由で変更した場合には、該当するアプリケーションプログラムを利用しているアプリケーションプログラム連携処理手続きの中のコンピュータを指定している部分を変更しなければならない。そのため、分散処理システムを構成するコンピュータやネットワーク、アプリケーションプログラムの変更に対する柔軟性に欠ける。

【００１７】本発明の目的は、個々の利用者が自分自身の利用方法に適するアプリケーションプログラム間の連携処理を行うことを可能とするアプリケーションプログラム間連携処理方法を提供することにある。

【００１８】本発明の他の目的は、データやメッセージに加え、アプリケーションプログラム間の連携処理手続き、データやメッセージを受け取ることにより、受信側アプリケーションプログラムで行われる処理手続き、受信側アプリケーションプログラムの提供する複数の機能を組み合わせた一連の処理手続きをアプリケーションプログラム間で送受信し、受信したこれら手続きを実行できる連携インタフェースをアプリケーションプログラムに付与することにより、アプリケーションプログラム間の連携処理手続きの変更を柔軟、かつ開発効率を低下させることなく行うことのできるアプリケーションプログラム間連携処理方法を提供することにある。

【００１９】

【課題を解決するための手段】上記目的を達成するために、本発明は、コンピュータネットワークに接続された複数のコンピュータ上で動作する複数のアプリケーションプログラム間の連携処理方法において、前記複数のアプリケーションプログラムの利用順序と各アプリケーションプログラムの利用手続きを記述した作業指示書を作成し、該作業指示書を、前記コンピュータネットワークを介して、目的のアプリケーションプログラムを有する複数のコンピュータへ順次転送し、前記複数のアプリケーションプログラムの各々には、前記作業指示書に記載された当該アプリケーションプログラムの利用手続きに応じて当該アプリケーションプログラムとの連携をとる連携インタフェースを設けておき、各連携インタフェースは、前記作業指示書を受けたとき、該連携インタフェースが当該作業指示書に記載された自己のアプリケーションプログラムに関する利用手続きに応じて自己のアプリケーションプログラムを起動し、目的とする制御および／またはデータの授受を行い、必要なアプリケーションプログラムを有するコンピュータを作業指示書が一巡することにより前記複数のアプリケーションプログラムによる一連の作業を実行するようにしたものである。

【００２０】このアプリケーションプログラム間連携処理方法において、前記作業指示書に記載された利用手続きの構成要素として、例えば、当該作業指示書が利用する全てのアプリケーションプログラムで使用されるデータ項目の一覧を表す全データ項目、各アプリケーションプログラムへ与えるべきデータが格納されている出力データ項目、各アプリケーションプログラムの処理結果が格納される入力データ項目、前記出力データ項目に格納されているデータを当該アプリケーションプログラムへ出力する手続きを表すデータ出力手続き、前記入力データ項目ごとのデータの入力手続きを表すデータ入力手続き、該データ入力手続きを用いて入力されたデータを処理する入力データ処理手続き、当該アプリケーションプログラムの実行を制御する制御手続き、のうち少なくとも１つを含む。

【００２１】好ましくは、各コンピュータ上に移動制御手段を有し、該移動制御手段が、自己のコンピュータ上の前記連携インタフェース及び他のコンピュータとの間の通信路を確保、管理し、該通信路を通して受信した前記作業指示書に記載された利用順序に従って当該作業指示書の移動を制御する。

【００２２】前記作業指示書の構成要素である制御手続きとして、例えば、該作業指示書を受信した前記連携インタフェースのアプリケーションプログラムを起動する起動手続き、該アプリケーションプログラムの実行を終了する終了手続き、該アプリケーションプログラムの有する１つ以上のコマンドを実行する実行手続き、それらを組み合せた制御スクリプトの少なくとも１つの手続きを有する。

【００２３】前記作業指示書の構成要素として、新たな作業指示書を生成するための生成手続きを有してもよい。前記利用手続きは、例えば、プログラムおよび関数名のいずれかであり、関数名の場合には前記連携インタフェースにおいて当該関数名に対応するプログラムを取得する。

【００２４】前記作業指示書に記述された利用順序は、コンピュータ名と、そのコンピュータ上で動作するアプリケーションプログラムおよび連携インタフェースからなるサービスエンティティの名称との組を該サービスエンティティの利用の順序に従って指定したものであり、前記コンピュータ名を省略可能とすることができる。

【００２５】さらに具体的には、本発明では、アプリケーションプログラム間連携処理を、アプリケーションプログラム間連携処理に関わる個々のアプリケーションプログラム本体にアプリケーションプログラム間連携処理手続きを実行するための連携インタフェースを付与したサービスエンティティ、連携させるサービスエンティティ間の移動先を表す利用順序とサービスエンティティの利用手続きを有する作業指示書（サービスエージェント）、このサービスエージェントをそれが有する利用順序に記述された移動先に移動させる通信路を生成、維持、管理する移動制御手段（ルーティングマネージャ）から構成する。

【００２６】サービスエンティティは、１つのアプリケーションプログラムの実行単位として分散処理システムを構成する個々のコンピュータ上に配置され、実行される。サービスエンティティを構成する連携インタフェースは、サービスエージェントの有する制御手続き、入出力データ手続き、表示手続きの中から現在のサービスエンティティに該当する手続きを選択し、その手続きを実行することにより、アプリケーションプログラム本体を実行制御する機能を持つ。また、実行する手続きに対応するプログラムが他のコンピュータ上にある場合には、そのプログラムをサービスエンティティが動作しているコンピュータ上にコピーし、そのプログラムを実行することにより、サービスエージェントの要求する手続きを実行する機能を持つ。

【００２７】アプリケーションプログラム間の連携処理を実現するサービスエージェントは、連携処理においてアプリケーションプログラムを連携させる順序を該当するアプリケーションプログラム本体を有するサービスエンティティの利用順序として記述した利用順序と、各サービスエンティティにおいて使用するアプリケーションプログラムの有する機能を利用する制御手続き、その制御手続きを用いてサービスエンティティの有するアプリケーションプログラムに実際に処理を行わせるために必要なデータを格納するデータ項目、データ項目に格納されたデータをアプリケーションプログラムに渡し、その処理結果データを該当するデータ項目に格納する入出力

データ処理手続き、サービスエージェントの有するデータをグラフィカルユーザインタフェースを通して表示するため表示手続きからなる利用手続きを有する。

【００２８】ルーティングマネージャは、分散処理システムを構成する各コンピュータ上に１つずつ配置され、動作しており、各コンピュータ上で動作するルーティングマネージャ間でサービスエージェントが移動するための通信路を構築し、各コンピュータ上で動作するサービスエンティティの名称とそのサービスエンティティとの間でサービスエージェントを移動させるための通信路を管理する機能を持つ。さらに、他のルーティングマネージャから通信路を通して受け取ったサービスエージェントの必要としているサービスエンティティが自コンピュータ上で利用できるか否かを判定し、利用できる場合にはサービスエージェントをその必要としているサービスエンティティに送り、利用できない場合にはサービスエージェントを他のコンピュータ上のルーティングマネージャに移動させる機能を持つ。これらの機能を用いて、サービスエージェントの利用順序に従って、利用順序に指定されたサービスエンティティにサービスエージェントを移動させることにより、サービスエージェントが連携処理のために必要とするサービスエンティティ間の移動を行う。

【００２９】

【作用】本発明では、アプリケーションプログラム間連携処理方法を上記の構成とすることにより、アプリケーションプログラム間の連携処理を作業指示書（サービスエージェント）というアプリケーションプログラム本体とは独立した形式で与えることができる。また、アプリケーションプログラム本体に連携処理のための連携インタフェースを付加し、この連携インタフェースがサービスエージェントの持つ利用手続きを実行することによりアプリケーションプログラム間の連携処理手続きをサービスエージェントを通して容易に変更することができる。

【００３０】分散処理システムを構成する各コンピュータ上で動作する移動制御手段としてのルーティングマネージャは、サービスエージェントを送受信するための通信路を確立し、ルーティングマネージャの動作しているコンピュータ上で動作するサービスエンティティの名称とそのサービスエンティティとの間でサービスエージェントを送受信するための通信路を管理する。サービスエンティティが起動されると、サービスエンティティ（アプリケーションプログラム＋連携インタフェース）の動作するコンピュータ上のルーティングマネージャとの間にサービスエージェントを送受信するための通信路を確立し、ルーティングマネージャにサービスエンティティの名称を送り、これを受けるルーティングマネージャは送られたサービスエンティティの名称とそのサービスエンティティへの通信路を管理する。各ルーティングマネージャは、他のコンピュータ上で動作しているルーティングマネージャから通信路を通して受け取ったサービスエージェントの利用順序に指定されたサービスエンティティがそれ自身の動作するコンピュータ上で利用できるか否かを判定し、利用できる場合にはサービスエージェントを該当するサービスエンティティに送り、利用できない場合にはサービスエージェントを他のコンピュータ上のルーティングマネージャに移動させる。

【００３１】サービスエンティティは、通信路を通してサービスエージェントを受け取ると、連携インタフェースがそのサービスエージェントの持つ利用手続きの中からそのサービスエンティティで使用される手続きとデータを取得し、取得した手続きを実行することにより、サービスエンティティの有するアプリケーションプログラム本体の処理を制御し、サービスエージェントの必要としている処理結果を得、その結果をサービスエージェントに送り、サービスエージェント中のデータ項目に格納する。サービスエージェントから取得した手続きに対応するプログラムが他のコンピュータ上にあることが指定されている場合には、連携インタフェースはそのプログラムをサービスエンティティが動作しているコンピュータ上にコピーし、そのプログラムを実行することにより、サービスエージェントの要求する手続きを実行する。サービスエージェントの必要としている処理結果を格納した後、サービスエンティティの連携インタフェースはサービスエージェントの移動先を利用順序にしたがって更新し、通信路を通じてルーティングマネージャにサービスエージェントを送る。ルーティングマネージャはサービスエンティティから送られたサービスエージェントの利用順序に指定されたサービスエンティティがそれ自身の動作するコンピュータ上で利用できるか否かを判定し、利用できる場合にはサービスエージェントを該当するサービスエンティティに送り、利用できない場合にはサービスエージェントを他のコンピュータ上のルーティングマネージャに移動させる。

【００３２】このようにルーティングマネージャは、サービスエージェントの利用順序に従って分散処理システムを構成するコンピュータ間で利用順序に指定されたサービスエンティティが動作するコンピュータにサービスエージェントを移動させ、サービスエンティティは受け取ったサービスエージェントの持つ手続きを連携インタフェースが取得し、実行することにより、アプリケーションプログラム間の連携処理を行う。

【００３３】本発明では、このようにアプリケーションプログラム間の連携処理手続きをサービスエージェントというアプリケーションプログラム本体とは独立した形式で与え、アプリケーションプログラム本体に連携処理のための連携インタフェースを付加し、この連携インタフェースがサービスエージェントの持つ利用手続きを実行することによりアプリケーションプログラム間の連携

処理手続きをサービスエージェントを通して容易に変更することができる。

【００３４】この連携処理方法を用いることにより、業務に応じて、その業務を進めるために必要なさまざまなアプリケーションプログラムとその操作を組み合わせた利用手順を、サービスエージェントの中に一連の作業手順と各作業で使用されるアプリケーションプログラムの利用手続きとして記述し、このサービスエージェントをコンピュータネットワークに接続されたコンピュータ上で動作している各種サービスエンティティの間で移動させ、サービスエージェントを受信したサービスエンティティはサービスエージェントに記述された利用手続きを実行することにより、従来、作業者が行っていた複数のアプリケーションプログラムを組み合わせた利用や、各作業者に依頼して行っていたアプリケーションプログラムの操作、及びその結果の取得を自動化するとともに、一連の作業に必要なアプリケーションプログラムの利用手順をシステム化することにより、各作業者の作業負荷を軽減し、作業の流れを円滑にすることができる。

【００３５】なお、本発明における好適なアプリケーションプログラムとしては、生産スケジュールの作成プログラム、生産システムの動作シミュレーションプログラムなどが考えられる。それぞれがコンピュータによって制御されている複数の製造設備から構成される工場の生産工程の管理を行なう際は、工場内の幾つかの生産物（ロット）に対し、次にどの設備で、どのような加工作業を行なわせるかを的確に指示する必要があり、従来はこのような指示を工程係が、時々刻々と変わる生産進行状況を判断して行なわなければならず、煩雑なうえ効率が悪いという問題があった。本発明によれば、上のような判断や指示をサービスを提供する設備とそれを受けるロットの間で情報を交換しながら自律的に（人手に依らず）行なうことが可能であり、以上のような生産システムの自律化などに好適である。また、本発明は、コンピュータネットワークを介して行う種々のサービス（オンラインショッピング、切符の予約など）への適用も可能である。

【００３６】

【実施例】以下では、図面を使用して本発明の実施例について詳細に説明する。本実施例では、アプリケーションプログラム本体と連携インタフェースから構成される複数のサービスエンティティの間をサービスエージェントが移動しながら、サービスエンティティを通してアプリケーションプログラムを連携させ、処理を進めていくアプリケーションプログラム間連携処理方法について説明する。

【００３７】図１は、本実施例のアプリケーションプログラム間連携処理方法の動作する分散処理システムの構成を表した図である。本実施例では、上記の分散処理システムにおいて、利用者が或る作業を行う際に、アプリケーションプログラムＡＰ１→ＡＰ２→ＡＰ３もしくはＡＰ４をこの順序で連携して処理させる必要があるものとする。また、各アプリケーションプログラムの処理においては、ＡＰ１はサービスエージェント（作業指示書：図３により後述）のデータ項目の項目１、項目２に格納されたデータを用いて処理を行い、その処理結果データを項目６に格納し、ＡＰ２は項目３、項目４に格納されたデータを用いて処理を行い、その処理結果データを項目７に格納し、ＡＰ３は項目４、項目６に格納されたデータを用いて処理を行い、その処理結果データを項目８に格納し、ＡＰ４は項目５に格納されたデータを用いて処理を行い、その処理結果データを項目９に格納するものとする。

【００３８】本分散処理システムは、コンピュータ（１）、コンピュータ間でデータ通信を行うためのネットワーク（０）、各コンピュータ上で動作するアプリケーションプログラム本体（３）、他のアプリケーションプログラムと連携して処理を行うための連携インタフェース（３０）、アプリケーションプログラム本体（３）に連携インタフェース（３０）を付与したアプリケーションプログラム間連携処理の基本となる処理単位であるサービスエンティティ（４）、各アプリケーションプログラム間を移動し連携処理を実現するサービスエージェント（２０）、サービスエージェント（２０）の移動する通信路及びサービスエンティティとの間の通信路を確立、管理し、サービスエージェント（２０）をサービスエンティティ（４）間で移動させるルーティングマネージャ（１０）から構成される。

【００３９】本実施例の説明上、コンピュータ（１）の名称をそれぞれＣ１、Ｃ２、Ｃ３とし、各アプリケーションプログラム本体（３）の名称をそれぞれＡＰ１、ＡＰ２、ＡＰ３、ＡＰ４とし、各アプリケーションプログラム本体ＡＰ１～ＡＰ４を有するサービスエンティティの名称をそれぞれＳＥ１、ＳＥ２、ＳＥ３、ＳＥ４とし、コンピュータＣ１～Ｃ３を区別する必要のある場合にはそれぞれ１ａ、１ｂ、１ｃの符号を、アプリケーションプログラム本体ＡＰ１～ＡＰ４を区別する必要のある場合にはそれぞれ３ａ、３ｂ、３ｃ、３ｄの符号を、サービスエンティティＳＥ１～ＳＥ４を区別する必要のある場合にはそれぞれ４ａ、４ｂ、４ｃ、４ｄの符号を用いる。

【００４０】図２にコンピュータ（１）の構成例を示す。コンピュータ（１）は、キーボード（１１）やマウス（１２）などから構成される入力装置（１３）、ＣＰＵ（１４）やメモリ（１５）を格納したコンピュータ本体である処理装置（１６）、データを表示する表示装置（１７）、データやアプリケーションプログラムを格納する外部記憶装置（１８）、データを印刷するプリンタ（１９）から構成される。

【００４１】入力装置（１３）は、上記以外のタブレッ

ト、タッチパネルでもよい。入力装置（１３）には、画像データを入力するためのスキャナ、音声を入力するためのマイクが加えられてもよい。入力装置（１３）のマウスは、表示装置（１７）上の位置を指定したり、表示装置（１７）に表示されたいくつかの選択肢を含むメニューの中から選択対象を指定するための手段であり、このような指定手段を有する他の装置、例えば光学式ペンやタッチパネルであってもよい。表示装置（１７）として、音声を出力するためのスピーカが加えられてもよい。

【００４２】ネットワーク（０）は、複数のコンピュータ（１）間でデータを送受信する手段であり、特定の場所内のネットワークである LAN（ Local Area Network ）、各拠点間のネットワークである WAN（ Wide Area Network ）、ネットワーク同士を相互に接続したネットワークであるインターネットのいずれのネットワークであってもよい。

【００４３】図１に示したルーティングマネージャ（１０）は、各コンピュータごとに１つずつ存在し、コンピュータ起動時に起動される。ルーティングマネージャ（１０）をコンピュータ起動時に起動する方法としては、例えば、ＵＮＩＸのデーモンを用いる方法などがある。

【００４４】本実施例によるアプリケーションプログラム間連携処理方法では、上記のネットワーク（０）に接続されたコンピュータ（１ａ）で作成されたサービスエージェント（２０）がルーティングマネージャ（１０）を通して、ネットワーク（０）に接続されたコンピュータ（１）間を移動し、これらのコンピュータ（１）上で動作するサービスエンティティ（４）の連携インタフェース（３０）を通してその中のアプリケーションプログラム本体（３）の処理を実行、制御し、アプリケーションプログラム間の連携処理を行う。このように、サービスエージェント（２０）を用い複数のサービスエンティティ（４）を通してアプリケーションプログラム本体（３）の実行の制御、管理を行い、一連の作業に用いられるアプリケーションプログラム間を連携して処理することにより、一連の作業を自動化、支援する。

【００４５】本実施例によるアプリケーションプログラム間連携処理方法では、一台以上のコンピュータ（１）上で動作する２つ以上のアプリケーションプログラム本体（３）を有するサービスエンティティ（４）で利用することが可能であるが、図１では３台のコンピュータと各々のコンピュータ上で動作する４つのサービスエンティティ（４）を、サービスエージェント（２０）を用いて連携させることを例にして説明する。このように、一台のコンピュータ（１）上には、複数のサービスエンティティ（４）を含むことができる。コンピュータ（１）やアプリケーションプログラム本体（３）を有するサービスエンティティ（４）の数は、図示の構成に限定されるものではない。

【００４６】本実施例によるアプリケーションプログラム間連携処理方法で用いられる典型的なサービスエージェント（２０）の構造を図３に示す。異なる複数個のサービスエージェント（２０）が同時にシステム内を流れうる。サービスエージェント（２０）は、個々の利用者がいずれかのコンピュータ上で作成する。また、これに加えて、作成されたサービスエージェント自体が、必要に応じて新たなサービスエージェントを生成することもある。

【００４７】図３に示すように、サービスエージェント（２０）は、個々のサービスエージェント（２０）を区別するための識別子（２０１）、サービスエージェントの移動先の一覧を記述した移動先リスト一覧（２０２）、サービスエージェントが移動してきた経路を管理する移動リスト一覧（２０３）、現時点でサービスエージェント（２０）が必要としているアプリケーションプログラム本体（３）を持つサービスエンティティ（４）の名称とそのサービスエンティティ（４）が動作しているコンピュータの名称を示す移動先名（２０４）、連携処理を行う複数のサービスエンティティ（４）が必要とする全てのデータ項目（２１１～２１９）を列挙した全データ項目（２１）、全データ項目（２１）の中の各データ項目（２１１～２１９）に応じた入力方法を記述したデータ入力手続き（２２）、全データ項目（２１）の中の各データ項目やサービスエンティティに応じた出力方法を記述したデータ出力手続き（２３）、データ入力手続き（２２）を用いて入力されたデータを処理する方法である入力データ処理手続き（２４）、このサービスエージェント（２０）の移動先を決定し、決定した移動先にサービスエージェント（２０）を送るための方法である移動手続き（２５）、各サービスエンティティ（４）の持つアプリケーションプログラム本体（３）の実行を制御するための手続きである制御手続き（２６）、新たなサービスエージェントを生成するための手続きである生成手続き（２７）から構成される。

【００４８】サービスエージェント（２０）の構造のうち、識別子（２０１）、移動先リスト一覧（２０２）、移動リスト一覧（２０３）、移動先名（２０４）は、サービスエージェント（２０）の作成時に必ず作成される項目である。上記以外の、全データ項目（２１）、データ入力手続き（２２）、データ出力手続き（２３）、入力データ処理手続き（２４）、移動手続き（２５）、制御手続き（２６）、生成手続き（２７）の各項目や手続きは、サービスエージェント（２０）を用いて行われるサービスエンティティ（４）間の連携処理の方法ごとに異なる要素であり、全ての要素が必ず存在するわけではない。

【００４９】全データ項目（２１）は、何等かの値を持つデータ項目（２１１～２１５）である入力済みデータ

項目（２７０）と、値を持たないデータ項目（２１６～２１９）である未入力データ項目（２８０）とから構成される。入力済みデータ項目（２７０）や未入力データ項目（２８０）は、サービスエージェント（２０）が或るサービスエンティティ（４）に受け取られた時に、そのサービスエンティティ（４）に渡すべきデータを格納しているデータ項目である出力データ項目（２７１）、或るサービスエンティティ（４）の実行結果データを格納するデータ項目である入力データ項目（２８１）を含んでいる。なお、ここでの入力および出力の別は、サービスエージェント（２０）から見たものであり、サービスエージェント（２０）からサービスエンティティ（４）へのデータの受け渡しを出力とし、その逆を入力としている。

【００５０】例えば、アプリケーションプログラム本体ＡＰ１が実行時に必要とするデータが項目１（２１１）、項目２（２１２）に格納されており、その実行結果データが項目６（２１６）に格納されるとすると、サービスエンティティＳＥ１では、項目１（２１１）と項目２（２１２）が出力データ項目（２７１）、項目６（２１６）が入力データ項目（２８１）である。アプリケーションプログラムＡＰ２が実行時に必要とするデータが項目３（２１３）、項目４（２１４）に格納されており、その実行結果データが項目７（２１７）に格納されるとすると、サービスエンティティＳＥ２では、項目３（２１３）と項目４（２１４）が出力データ項目、項目７（２１７）が入力データ項目である。同様に、アプリケーションプログラム本体ＡＰ３が実行時に必要とするデータが項目４（２１４）、項目６（２１６）に格納されており、その実行結果データが項目８（２１８）に格納されるとすると、サービスエンティティＳＥ３では、項目４（２１４）と項目６（２１６）が出力データ項目（２７１）、項目８（２１８）が入力データ項目（２８１）であり、アプリケーションプログラム本体ＡＰ４が実行時に必要とするデータが項目５（２１５）に格納されており、その実行結果データが項目９（２１９）に格納されるとすると、サービスエンティティＳＥ４では、項目５（２１５）が出力データ項目（２７１）、項目９（２１９）が入力データ項目（２８１）となる。このように出力データ項目（２７１）や入力データ項目（２８１）は、サービスエージェント（２０）が受け取られたサービスエンティティ（４）ごとに異なる。

【００５１】データ入力手続き（２２）には、未入力データ項目（２８０）の中からサービスエンティティ（４）の持つアプリケーションプログラム本体（３）に応じた入力データ項目（２８１）を選択するための手続きである入力データ選択手続き（２２１）、利用者により画面から入力された値を取得するための手続きである入力データ取得手続き（２２２）、アプリケーションプログラム本体（３）の出力結果データを取得するための手続きであるデータ取得手続き（２２３）、このデータ取得手続き（２２３）を用いて取得されたアプリケーションプログラム本体（３）の出力結果データを入力データ項目（２８１）のデータ形式に変換するための手続きである入力データ変換手続き（２２４）、入力データ取得手続き（２２２）を用いて得られた値や入力データ変換手続き（２２４）を用いて入力データ項目（２８１）のデータ形式に変換された値を入力データ項目（２８１）に格納するための手続きである入力データ格納手続き（２２５）がある。

【００５２】ここでいう「データ形式の変換」とは、データの書式の変換を意味する。具体的な例として、或るデータが属性名「個数」、「単価」を持ち、「個数」の値として「１０」、「３０」、及び「単価」の値として「２０」、「４０」を、アプリケーションプログラムが表１に示すような「アプリケーションプログラムの実行結果の出力書式」で出力し、入力データ項目には表２に示すような「サービスエージェントの入力データ項目の書式」で格納される場合、入力データ変換手続きは、以下のように、「アプリケーションプログラムの実行結果の出力書式」から「サービスエージェントの入力データ項目の書式」への書式の変換を行なう。

【００５３】

【表１】

・「アプリケーションプログラムの実行結果の出力書式」

```
（（（個数　単価）
　　（１０　２０））
　（（個数　単価）
　　（３０　４０））
```

【００５４】

【表２】

・「サービスエージェントの入力データ項目の書式」

```
（（個数　単価）
　（１０　２０）
　（３０　４０））
```

【００５５】データ出力手続き（２３）には、入力済みデータ項目（２７０）の中からサービスエンティティ（４）の持つアプリケーションプログラム本体（３）に応じた出力データ項目（２８１）を選択するための手続きである出力データ選択手続き（２３１）、この出力データ選択手続き（２３１）を用いて選択された出力データ項目（２８１）の値をアプリケーションプログラム本体（３）のデータ入力形式に変換するための手続きである出力データ変換手続き（２３２）、この出力データ変

換手続き（２３２）を用いて変換された出力データ項目（２８１）の値をアプリケーションプログラム本体（３）に渡すための手続きであるパラメタ設定手続き（２３３）、出力データ変換手続き（２３２）を用いて変換された出力データ項目（２８１）の値を画面に表示するための手続きである出力データ表示手続き（２３４）がある。

【００５６】入力データ処理手続き（２４）は、入力データ取得手続き（２２２）やデータ取得手続き（２２３）を用いて取得されたデータを処理するための手続きである。例えば、入力データ取得手続き（２２２）を用いて取得された複数のデータの和を計算する処理などがこの入力データ処理手続き（２４）として記述される。

【００５７】移動手続き（２５）は、サービスエージェント（２０）の移動先を決定し、決定した移動先にサービスエージェント（２０）を送るための手続きである。移動手続き（２５）の移動先の決定方法には、サービスエージェント作成時に与えられた移動順序に従って次の移動先を決定する方法、入力済みデータ項目（２７０）や未入力データ項目（２８０）の中の任意のデータ項目の組み合わせに応じてサービスエージェントの移動先を決定する方法がある。すなわち、サービスエージェント作成時に与えられた移動順序に従って次の移動先を決定する場合は、サービスエージェント（２０）の移動先リスト一覧（２０２）の先頭要素を取り出し、それを移動先名（２０４）に格納し、移動先リスト一覧（２０２）の値を更新し、更新したサービスエージェント（２０）を後述する入出力チャネル（４０３）を通してルーティングマネージャ（１０）に送る。任意のデータ項目の組合せに応じて移動先を決定する場合は、図２１に示すように、着目するデータ項目の値を調べ、その値により決定される移動先を移動先名（２０４）に格納し、移動先名を更新したサービスエージェント（２０）を入出力チャネル（４０３）を通してルーティングマネージャ（１０）に送る。

【００５８】制御手続き（２６）は、サービスエンティティ（４）の連携インタフェース（３０）で実行されるアプリケーションプログラム本体（３）の実行を制御するための手続きであり、アプリケーションプログラム本体（３）の起動・終了、アプリケーションプログラム本体（３）のコマンド実行、及び複数のコマンドの組み合わせである制御スクリプトがある。

【００５９】生成手続き（２７）は、新たなサービスエージェント（２０）を生成するための手続きであり、サービスエージェントを生成する制御スクリプトがある。

【００６０】図４～図６に、各コンピュータ上で動作するルーティングマネージャ（１０）により構築される論理的な通信路の接続形態を示す。ルーティングマネージャ（１０）は、複数のコンピュータ間の論理的通信路の確立と維持、各サービスエンティティ（４）との間の入出力チャネル（図６：４０３）の確立と維持を行う。論理的通信路は、相互に接続された２つのルーティングマネージャ（１０）を接続する論理的通信路であるチャネル（４０）の組み合わせから構成される。図４では、コンピュータＣ１、Ｃ２、Ｃ３上で動作する３つのルーティングマネージャ（１０）ＲＭ１、ＲＭ２、ＲＭ３の間で、ＲＭ１とＲＭ２、ＲＭ２とＲＭ３が接続され、ＲＭ１－ＲＭ２－ＲＭ３という論理的通信路を構成していることを表している。図５は、図４に示した論理的通信路を構成しているルーティングマネージャ（１０）ＲＭ２に新たなルーティングマネージャ（１０）ＲＭ４が接続され、論理的通信路の構成が変化した状態を示す図である。図６は、或る一台のコンピュータ内におけるルーティングマネージャ（１０）ＲＭ３とサービスエンティティ（４）ＳＥ３、ＳＥ４が入出力チャネル（４０３）により接続された接続状態、及びサービスエンティティ（４）を構成する連携インタフェース（３０）とアプリケーションプログラム本体（３）の間がアプリケーションインタフェース（３８）で接続された接続状態を表している。入出力チャネル（４０３）は、ルーティングマネージャ（１０）とサービスエンティティ（４）を接続した通信路であり、実際にはサービスエンティティ（４）内の連携インタフェース（３０）とルーティングマネージャ（１０）の間を接続する。アプリケーションインタフェース（３８）は、サービスエンティティ（４）内に生成される連携インタフェース（３０）とアプリケーションプログラム本体（３）の入出力を接続するインタフェースである。

【００６１】ルーティングマネージャ（１０）は、チャネル（４０）を接続するための受け口となるチャネルポート（４０１）を持つ。チャネルポートは個々のチャネルポート（４０１）を識別するためのチャネルポート番号を持つ。チャネルポート番号は、ルーティングマネージャ（１０）により、チャネルポート（４０１）生成時に与えられる。すなわち、チャネル（４０）とは２つのルーティングマネージャ（１０）のチャネルポート（４０１）を接続した通信路であり、チャネル（４０）を生成するとは、任意の２つのルーティングマネージャ（１０）のチャネルポート（４０１）の間を接続することである。

【００６２】ルーティングマネージャ（１０）は、チャネル（４０）を生成することによりチャネルポート（４０１）が使用されると新たなチャネルポート（４０２）を１つ生成する。そのためルーティングマネージャ（１０）は、常に他と接続されていない空きチャネルポート（４０２）を持つ。この空きチャネルポート（４０２）は、他のルーティングマネージャ（１０）から接続要求がなされた場合にそのルーティングマネージャ（１０）との間でチャネル（４０）を生成するために使用される。

【００６３】ルーティングマネージャ（１０）が生成した空きチャネルポート（４０２）は、サービスエンティティ（４）とルーティングマネージャ（１０）を接続する際にも、ルーティングマネージャ（１０）同士を接続する場合と同様に用いられる。すなわち、ルーティングマネージャ（１０）とサービスエンティティ（４）との間に入出力チャネル（４０３）を生成する場合には、サービスエンティティ（４）が起動されたときに新たな空きチャネルポートをサービスエンティティ（４）の連携インタフェース（３０）に生成し、サービスエンティティ（４）からルーティングマネージャ（１０）に接続要求を行い、ルーティングマネージャ（１０）が空きチャネルポート（４０２）とサービスエンティティ（４）の連携インタフェース（３０）の空きチャネルポートの間に入出力チャネル（４０３）を作成し、ルーティングマネージャ（１０）のみが新たな空きチャネルポート（４０２）を生成する。ルーティングマネージャ（１０）は、このようにチャネル（４０）が生成されると、新たな空きチャネルポート（４０２）を１つ生成し、常に１つの空きチャネルポート（４０２）が残るようにチャネルポートの管理をする。

【００６４】サービスエンティティ（４）は、起動時にルーティングマネージャ（１０）との間で入出力チャネル（４０３）を生成するための空きチャネルポート（４０２）を連携インタフェース（３０）に生成し、ルーティングマネージャ（１０）との間で入出力チャネル（４０３）を生成することに加え、連携インタフェース（３０）とアプリケーションプログラム本体（３）の入出力を接続するためのアプリケーションインタフェース（３８）を持ち、このアプリケーションインタフェース（３８）を用いて連携インタフェース（３０）とアプリケーションプログラム本体（３）の間の入出力、すなわち連携インタフェース（３０）からアプリケーションプログラム本体（３）にコマンドを送ったり、アプリケーションプログラム本体（３）の実行結果データを連携インタフェース（３０）に受け取ったりすること、を行う。

【００６５】チャネルポート（４０１）は、受信時のバッファリング機能を持ち、チャネル（４０）や入出力チャネル（４０３）を通して送られてきたデータを、読みだし要求があるまで保持する。

【００６６】図７に、ルーティングマネージャ（１０）を実現するための処理装置（１６）におけるプログラムの構造を示す。ルーティングマネージャ（１０）は、処理装置（１６）のメモリ（１５）上におかれる管理テーブル作成プログラム（４１）、接続要求プログラム（４２）、接続管理プログラム（４３）、移動管理プログラム（４４）、接続待ちプログラム（４５）の各プログラムと、接続先管理テーブル（４６）、サービスエンティティ管理テーブル（４７）、接続管理テーブル（４８）から構成される。

【００６７】処理装置（１６）のメモリ（１５）上におかれたルーティングマネージャ（１０）が起動されると、まず管理テーブル作成プログラム（４１）が外部記憶装置（１８）に格納されている接続先コンピュータの名称の一覧を読み込み、処理装置（１６）のメモリ（１５）上に接続先管理テーブル（４６）を作成する。次にルーティングマネージャ（１０）の動作するコンピュータで動作するサービスエンティティの名称の一覧を読み込み、処理装置（１６）のメモリ（１５）上にサービスエンティティ管理テーブル（４７）を作成する。

【００６８】接続要求プログラム（４２）は、管理テーブル作成プログラム（４１）により処理装置（１６）のメモリ（１５）上に作成された接続先管理テーブル（４６）に格納されたコンピュータ名の一覧の先頭から順に接続先コンピュータ名を取得し、その接続先コンピュータに対して接続を要求する。接続ができない場合には次の接続先コンピュータ名を取得し、接続ができるまで順次接続要求を行う。接続先管理テーブル（４６）に格納された接続先コンピュータ名がなくなるまで、順次、接続要求を行っても接続できない場合、エラーメッセージを出力し、処理を終了する。

【００６９】接続管理プログラム（４３）は、接続要求プログラム（４２）により接続されたコンピュータ（接続先コンピュータ）と接続が完了した旨の報告を受け取ると、接続要求を行ったコンピュータ（接続元コンピュータ）のコンピュータ名を接続先コンピュータに送信し、接続先コンピュータから接続先コンピュータ名が送信されてくるのを待つ。接続先コンピュータから送られた接続先コンピュータ名を受け取ると、その接続先コンピュータの名称と接続先コンピュータに接続されたチャネル（４０）のチャネルポート番号のペアを接続管理テーブル（４８）に格納する。後述する接続待ちプログラム（４５）がチャネルポート番号を受け取ったことにより制御が渡されてきた場合、そのチャネルポート番号から接続元コンピュータ名、もしくはサービスエンティティ（４）の名称が送られてくるのを待つ。接続元コンピュータ名、もしくはサービスエンティティ（４）の名称を受け取るとその名称とその間に接続された論理的通信路のチャネルポート番号のペアを接続管理テーブル（４８）に格納し、送られてきた名称が接続元コンピュータ名の場合にはその接続元コンピュータに対して接続先コンピュータ名を送信する。

【００７０】移動管理プログラム（４４）は、接続管理プログラム（４３）が接続元コンピュータ名またはサービスエンティティ（４）の名称とチャネルポート番号のペアを接続管理テーブル（４８）に格納するか、或る一定時間の間に接続要求がなされなかった場合に起動される。移動管理プログラム（４４）は、接続管理テーブル（４８）の先頭から順にチャネルポート番号を取得し、そのチャネルポート番号（４０１）に接続されたチャネ

ル（４０）からサービスエージェント（２０）を読み出し、読み出したサービスエージェント（２０）の移動先名（２０４）に指定されたサービスエンティティの名称が接続管理テーブル（４８）に格納されているか否かをチェックする。格納されている場合には、そのサービスエージェント（２０）をサービスエンティティ（４）に送る。格納されていない場合には、接続されている他のルーティングマネージャ（１０）を接続管理テーブル（４８）から検索し、その結果得られたルーティングマネージャ（１０）にそのサービスエージェント（２０）を送る。チャネル（４０）から読み出すサービスエージェント（２０）がない場合には、次のチャネルポート番号（４０１）に接続されたチャネル（４０）から同様にサービスエージェント（２０）を読み出す。これを接続管理テーブル（４８）に格納されたチャネルポート番号（４０１）がなくなるまで繰り返す。接続管理テーブル（４８）に格納されたチャネルポート番号の全てについて順次読み出しを行うと次の接続待ちプログラム（４５）に制御を渡す。

【００７１】接続待ちプログラム（４５）は、或る一定時間他のルーティングマネージャ（１０）もしくはサービスエンティティ（３０）からの接続要求を待つ。或る一定時間の間に接続要求があれば、その接続要求元のルーティングマネージャ（１０）との間でチャネル（４０）を接続するか、接続要求元のサービスエンティティ（３０）との間で入出力チャネル（４０３）を接続し、新たな空きチャネルポート（４０２）を生成し、新たに接続されたチャネル（４０）の接続されたチャネルポート（４０１）のチャネルポート番号を接続管理プログラム（４３）に送る。或る一定時間の間に接続要求がなければ、移動管理プログラム（４４）に制御を渡す。

【００７２】図８（ａ）に接続先管理テーブル（４６）の構造を、図８（ｂ）にサービスエンティティ管理テーブル（４７）の構造を、図８（ｃ）に接続管理テーブル（４８）の構造を示す。

【００７３】接続先管理テーブル（４６）は、接続先のコンピュータの名称の一覧を保持する接続先コンピュータ名フィールド（４６０）を持つ。接続先コンピュータ名フィールド（４６０）は、ルーティングマネージャ（１０）の起動時に接続要求を行うコンピュータの名称であるコンピュータ名（４６１）を値として持つ。この接続先管理テーブル（４６）には、当然ながら自己のコンピュータの情報は含まない。

【００７４】サービスエンティティ管理テーブル（４７）は、ルーティングマネージャ（１０）の動作するコンピュータ上で動作するサービスエンティティ（４）の名称の一覧を保持するサービスエンティティ名フィールド（４７０）を持つ。サービスエンティティ名フィールド（４７０）は、ルーティングマネージャ（１０）の動作するコンピュータ上で動作するサービスエンティティ（４）の名称（４７１）を値として持つ。

【００７５】接続管理テーブル（４８）は、チャネル（４０）により接続された他のルーティングマネージャ（１０）の動作するコンピュータの名称や入出力チャネル（４０３）により接続されたサービスエンティティ（４）の名称を保持する接続先名称フィールド（４８０）、それらの接続先との接続に用いられているチャネル（４０）や入出力チャネル（４０３）が接続されているチャネルポート（４０１）のチャネルポート番号を保持するチャネルポート番号フィールド（４８１）、接続先がルーティングマネージャ（１０）であるかサービスエンティティ（４）であるかを区別するための接続先種別フィールド（４８４）を持つ。接続先名称フィールド（４８０）は、ルーティングマネージャ（１０）にチャネル（４０）や入出力チャネル（４０３）を用いて接続されている接続先名（４８２）を値として持ち、チャネルポート番号フィールド（４８１）は、チャネル（４０）や入出力チャネル（４０３）が接続されているチャネルポート番号（４８３）を接続先名（４８２）に対応付けて持つ。接続先種別フィールド（４８４）は、チャネルの接続先がルーティングマネージャであることを示すルーティングマネージャ、サービスエンティティであることを示すサービスエンティティのどちらか一方の値を取る。この２つの値をあわせて接続先種別名（４８５）と呼ぶ。

【００７６】図９に連携インタフェース（３０）を実現するための処理装置（１６）におけるプログラムの構造を示す。連携インタフェース（３０）は、処理装置（１６）のメモリ（１５）上におかれるデータ作成プログラム（３１）、データ出力プログラム（３２）、実行制御プログラム（３３）、データ取得プログラム（３４）、データ格納プログラム（３５）、移動先操作プログラム（３６）、関数制御プログラム（３７）の各プログラムと、関数テーブル（３７１）とから構成される。関数テーブル（３７１）を設けることにより、サービスエージェント（２０）自体にプログラムを保持する必要がなくなり、コンピュータ間のサービスエージェント（２０）の伝送時の負荷を軽減することができる。

【００７７】連携インタフェース（３０）は、入出力チャネル（４０３）を通じてサービスエージェント（２０）を受け取ると、受け取ったサービスエージェント（２０）をデータ作成プログラム（３１）に渡す。

【００７８】データ作成プログラム（３１）は、受け取ったサービスエージェント（２０）の中の出力データ選択手続き（２３１）を用いて、入力済みデータ項目（２７０）の中からアプリケーションプログラム本体（３）に渡すデータを格納しているデータ項目である出力データ項目（２７１）を選択する。

【００７９】出力データ項目（２７１）が決定されると、サービスエージェント（２０）の中の出力データ変

換手続き（２３２）を用いて、出力データ項目（２７１）に格納されているデータからアプリケーションプログラム本体（３）に渡すデータ、もしくは画面に出力するデータを作成し、どちらのデータであるかのタグを付加し、データ出力プログラム（３２）に作成したデータを渡す。

【００８０】データ出力プログラム（３２）にデータが渡されると、データ出力プログラム（３２）は受け取ったデータのタグからデータを画面に出力するか、アプリケーションプログラム本体（３）に送るかを判定し、画面に出力する場合にはサービスエージェント（２０）の出力データ表示手続き（２３４）を用いて画面に出力する。アプリケーションプログラム本体（３）に渡す場合、データを実行制御プログラム（３３）に渡す。サービスエージェント（２０）に記述された出力データ表示手続き（２３４）が、プログラムではなく単なる関数名だけの場合、データ出力プログラム（３２）は、関数制御プログラム（３７）にその関数名を渡す。

【００８１】関数制御プログラム（３７）は、関数名を受け取ると、関数テーブル（３７１）からその関数名に該当する関数を検索し、検索したプログラムを実行する。実行制御プログラム（３３）は、データ出力プログラム（３２）からのデータを受け取ると、サービスエージェント（２０）の制御手続き（２６）を実行してアプリケーションプログラム本体（３）の実行を制御する。アプリケーションプログラム本体（３）を実行する際に、アプリケーションプログラム本体（３）にデータを渡す必要がある場合には、サービスエージェント（２０）のパラメタ設定手続き（２３３）を用いてアプリケーションプログラム本体（３）にデータを送る。アプリケーションプログラム本体（３）の実行を制御し、制御手続きの実行が終了すると、サービスエージェント（２０）のデータ取得手続き（２２３）を用いてアプリケーションプログラム本体（３）の実行結果データを取得し、取得したデータをデータ取得プログラム（３４）に渡す。制御手続き（２６）が関数名のみであったり、制御手続き（２６）の中に関数名のみの処理が含まれている場合には、その関数名を関数制御プログラム（３７）に渡し、該当する関数のプログラムを実行する。

【００８２】データ取得プログラム（３４）は、実行結果データを受け取るとサービスエージェント（２０）の入力データ選択手続き（２２１）を用いてサービスエージェント（２０）の全データ項目（２１）の中からアプリケーションプログラム本体（３）の実行結果データを格納するデータ項目である入力データ項目（２８１）を選択する。画面が出力されている場合には、サービスエージェント（２０）の入力データ取得手続き（２２２）を用いて画面に入力されたデータのうち、必要なデータのみを取得する。これらの取得したデータ（アプリケーションプログラム本体（３）の実行結果データ、画面からの入力データ）と入力データ項目（２８１）を、データ格納プログラム（３５）に渡す。サービスエージェント（２０）に記述された入力データ取得手続き（２２２）が、プログラムではなく単なる関数名だけの場合、データ取得プログラム（３４）は、関数制御プログラム（３７）にその関数名を渡し、該当する関数名の関数のプログラムを実行する。

【００８３】データ格納プログラム（３５）は、データ取得プログラム（３４）からのデータと入力データ項目（２８１）を受け取ると、サービスエージェント（２０）の入力データ処理手続き（２４）の領域を参照し、入力データ処理手続き（２４）があれば、その入力データ処理手続き（２４）を用いて、受け取ったデータ及び全項目データ（２１）に格納されているデータを処理し、処理後のデータを入力データ変換手続き（２２４）を用いて、受け取った入力データ項目（２８１）のデータ形式に変換し、変換したデータを入力データ項目（２８１）のそれぞれのデータ項目に格納する。入力データ処理手続き（２４）がない場合には、入力データ変換手続き（２２４）を用いて、受け取った入力データ項目（２８１）のデータ形式に変換し、変換したデータを入力データ項目（２８１）のそれぞれのデータ項目に格納する。

【００８４】移動先操作プログラム（３６）は、入力データ項目（２８１）にデータが格納されると呼び出され、移動先リスト一覧（２０２）の先頭要素を取りだしそれを移動先名（２０４）に格納し、このようにして移動先名（２０４）を更新したサービスエージェント（２０）を入出力チャネル（４０３）を通してルーティングマネージャ（１０）に送る。

【００８５】図１０（ａ）に関数制御プログラム（３７）の使用する関数テーブル（３７１）の構造を示す。関数テーブル（３７１）は、関数名フィールド（３７２）と関数の処理プログラム本体である処理フィールド（３７３）からなる。処理フィールド（３７３）は、関数の処理内容を記述したプログラム（３７３ａ）、または、関数の処理内容を記述したプログラムの格納位置（３７３ｂ）のいずれかが記述される。

【００８６】関数制御プログラム（３７）は、該当する関数名の処理フィールド（３７３）がプログラム（３７３ａ）であれば、そのプログラムを実行する。プログラムの格納位置（３７３ｂ）である場合、その格納位置にあるプログラム（３７４）をネットワーク（０）を通じて処理装置（１６）のメモリ（１５）上に読み込んだ（コピーした）後、そのプログラムを実行する。このような関数テーブルにプログラム自体ではなくプログラムの格納位置を保持しておくことにより、他のコンピュータ上にあるプログラムを共用することができる。これは、関数テーブルのメモリ容量を軽減するとともに、プログラムの変更時に１カ所のプログラムのみの変更で済

むという利点を有する。

【００８７】プログラムの格納位置（３７３ｂ）は、図１０（ｂ）の記述形式により指定される。格納位置は、プログラムの格納されているコンピュータの名称（３７５ａ）とコンピュータ名（３７５ａ）で指定されたコンピュータ上のプログラムの格納位置を表す格納パス（３７５ｃ）、コンピュータ名（３７５ａ）と格納パス（３７５ｃ）を区別するための区切り記号（３７５ｂ）を用いて、コンピュータ名（３７５ａ）、区切り記号：（３７５ｂ）、格納パス（３７５ｃ）の形式で指定する。

【００８８】図１１（ａ）にサービスエージェント（２０）の移動先リスト一覧（２０２）に格納される移動先リスト（９１）の記述形式を、その具体的記述例を図１１（ｂ）に示す。

【００８９】移動先リスト（９１）は、サービスエージェント（２０）を移動させる先のコンピュータの名称である移動先コンピュータ名（９１２）、移動先のコンピュータで利用するサービスエンティティ（４）の名称であるサービスエンティティ名（９１３）の組みである移動先（９１１）を、サービスエージェント（２０）の移動順序に従って矢印（→）でつないだものである。移動先のコンピュータ名が分からない場合には、移動先コンピュータ名（９１２）にＵｎｋｎｏｗｎを指定する。

【００９０】図１１（ｂ）の例では、サービスエージェント（２０）は、Ｃ１という名称のコンピュータのＳＥ１という名称のサービスエンティティ（４）の持つアプリケーションプログラム本体（３）ＡＰ１を用いて処理を行った後、Ｃ２という名称のコンピュータのＳＥ２という名称のサービスエンティティ（４）の持つアプリケーションプログラム本体（３）ＡＰ２を用いて処理を行い、次にＣ３という名称のコンピュータのＳＥ３という名称のサービスエンティティ（４）の持つアプリケーションプログラム本体（３）ＡＰ３を用いて処理を行い、最後にサービスエンティティ（４）ＳＥ４の動作するコンピュータ名は不明であるが、ＳＥ４という名称のサービスエンティティ（４）の持つアプリケーションプログラム本体（３）ＡＰ４を用いて処理を行うことを示している。

【００９１】図１２（ａ）にサービスエージェント（２０）の移動リスト一覧（２０３）に格納される移動リストの記述形式（１０１）を、その具体的記述例を図１２（ｂ）に示す。

【００９２】移動リスト一覧（２０３）は、サービスエージェント（２０）が利用するサービスがネットワーク上のどこのコンピュータ上で動作しているかを探して移動する際に用いる。移動リスト一覧（２０３）は、具体的には、図１１で示した移動先リストの移動先コンピュータ名（９１２）がＵｎｋｎｏｗｎの時に用いられる。移動先コンピュータ名（９１２）がＵｎｋｎｏｗｎの場合、各コンピュータのルーティングマネージャ（１０）は、サービスエージェント（２０）の移動先名（２０４）に指定されている名称を持つサービスエンティティ（４）が存在するか否かをチェックし、存在しない場合、そのコンピュータの名称を移動コンピュータ名（１０２）に格納し、他のコンピュータ上のルーティングマネージャ（１０）にサービスエージェント（２０）を送る。

【００９３】移動リスト（１０１）は、移動先コンピュータ名（９１２）がＵｎｋｎｏｗｎの場合に、サービスエージェント（２０）の移動先名（２０４）に指定されているサービスエンティティ（４）が存在しないコンピュータの名称をサービスエージェント（２０）が移動してきた順序に従って矢印（→）でつないだものである。移動リスト（１０１）は、サービスエージェント（２０）が、移動するたびにそのコンピュータ名が順次、追加されていき、サービスエージェント（２０）の移動先名（２０４）に指定されているサービスエンティティ（４）が動作するコンピュータが見つかった場合にクリアされる。このように、一度探索したコンピュータをこのリスト１０１に保存し、同一のコンピュータを何度も検索するのを防ぐことができる。移動リスト一覧（２０３）は、これが無い場合、エージェント（２０）が利用するサービスを探して同じコンピュータを何度も訪れる可能性があり、このような事態を回避して効率的なサービスの探索を行なう為には有用なものである。

【００９４】図１２（ｂ）の例では、サービスエージェント（２０）の移動先名（２０４）に指定されているサービスエンティティ（４）がＣ１という名称のコンピュータに存在せず、次にＣ２という名称のコンピュータにも存在しなかったことを示している。

【００９５】サービスエージェント（２０）を用いたアプリケーションプログラム間の連携処理方法を実現するためにルーティングマネージャ（１０）が行うサービスエージェント（２０）の移動処理手続きのフローを図１３～図１６に示す。

【００９６】サービスエージェント（２０）は、それが作成されたコンピュータのルーティングマネージャ（１０）に送られる。この時点で、サービスエージェント（２０）の移動先リスト一覧（２０２）の値は図１１（ｂ）に示した移動順序であるとする。この時のサービスエージェント（２０）の移動先名（２０４）の値は、移動先リスト一覧（２０２）の値である移動先リスト（９１）の先頭の要素ＳＥ１（９１１ａ）となる。

【００９７】ルーティングマネージャ（１０）は、サービスエージェント（２０）を受け取る（１１００）と、まずサービスエージェント（２０）の移動先名（２０４）の移動先コンピュータ名がルーティングマネージャ（１０）の動作しているのコンピュータ名と同じか否かをチェックする（１１０１）。

【００９８】サービスエージェント（２０）の移動先名

（２０４）の移動先コンピュータ名とルーティングマネージャ（１０）の動作しているコンピュータ名が異なる場合、接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がルーティングマネージャである接続先の中から、接続先名称フィールド（４８０）の接続先名（４８２）が移動先コンピュータ名と同じ接続先が登録されているか否かをチェックする（１１０２）。移動先コンピュータ名が登録されている場合、その移動先コンピュータ名に該当する接続先名（４８２）のチャネルポート番号フィールド（４８１）に格納されているチャネルポート番号（４８３）を取りだし（１１０３）、そのチャネルポート番号（４８３）に接続されているチャネル（４０）を通して移動先コンピュータのルーティングマネージャ（１０）にサービスエージェント（２０）を送る（１１０４）。登録されていない場合、接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がルーティングマネージャである接続先の中から、最初に登録されている接続先名（４８２）のチャネルポート番号フィールド（４８１）に格納されているチャネルポート番号（４８３）を取りだし（１１０５）、そのチャネルポート番号に接続されたチャネル（４０）を通してチャネルに接続されている登録先コンピュータのルーティングマネージャ（１０）にサービスエージェント（２０）を送る（１１０６）。

【００９９】サービスエージェント（２０）の移動先名（２０４）の移動先コンピュータ名（９１２）とルーティングマネージャ（１０）の動作しているコンピュータ名が同じである場合、接続管理テーブル（４８）の接続先種別フィールド（４８４）に格納されている接続先種別名（４８５）がサービスエンティティである接続先の中から接続先名称フィールド（４８０）の接続先名（４８２）が移動先名（２０４）のサービスエンティティ名（９１３）と同じサービスエンティティが登録されているか否かをチェックする（１１０７）。登録されていれば、その登録されているサービスエンティティのチャネルポート番号フィールド（４８１）に格納されているチャネルポート番号（４８３）を取得し（１１０８）、そのチャネルポート番号（４８３）に接続されている入出力チャネル（４０３）を通して、サービスエンティティ（４）の連携インタフェース（３０）にサービスエージェント（２０）を送る（１１０９）。

【０１００】サービスエンティティ名が接続管理テーブル（４８）に登録されていなければ、サービスエンティティ管理テーブル（４７）にサービスエンティティ名が登録されているか否かをチェックする（１１１０）。サービスエンティティ管理テーブル（４７）にサービスエンティティ名が登録されていれば、そのサービスエンティティ（４）を起動する（１１１１）。起動されたサービスエンティティ（４）の連携インタフェース（３０）は、ルーティングマネージャ（１０）に接続要求を行う。ルーティングマネージャ（１０）は、サービスエンティティ（４）の連携インタフェース（３０）からの接続要求を待ち（１１１２）、接続要求を受け取るとルーティングマネージャ（１０）と連携インタフェース（３０）間の入出力チャネル（４０３）を生成し（１１１３）、起動したサービスエンティティ名とその入出力チャネル（４０３）のチャネルポート番号を接続管理テーブル（４８）に登録し（１１１４）、生成した入出力チャネル（４０３）を通してサービスエージェント（２０）をサービスエンティティ（４）の連携インタフェース（３０）に送る（１１０９）。サービスエンティティ管理テーブル（４７）にサービスエンティティ名が登録されていなければ、サービスエージェント（２０）の移動先名（２０４）の移動先コンピュータ名（９１２）をＵｎｋｎｏｗｎに変更し（１１１５ａ）、接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がルーティングマネージャである接続先の中から、最初に登録されている接続先名（４８２）のチャネルポート番号フィールド（４８１）に格納されているチャネルポート番号（４８３）を取りだし（１１０５）、そのチャネルポート番号に接続されたチャネル（４０）を通してチャネルに接続されている登録先コンピュータのルーティングマネージャ（１０）にサービスエージェント（２０）を送る（１１０６）。

【０１０１】サービスエージェント（２０）の移動先名（２０４）の移動先コンピュータ名（９１２）がＵｎｋｎｏｗｎの場合、図１４に移り、移動先名（２０４）のサービスエンティティ名が、接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がサービスエンティティである接続先の中から接続先名称フィールド（４８０）の接続先名（４８２）がサービスエージェント（２０）の移動先名（２０４）のサービスエンティティ名と同じ接続先が登録されているか否かをチェックする（１１１５ｂ）。登録されていれば、その登録されているサービスエンティティのチャネルポート番号フィールド（４８１）に格納されているチャネルポート番号（４８３）を取得し（１１１６）、取得したチャネルポート番号（４８３）に接続されている入出力チャネル（４０３）を通して、サービスエンティティ（４）の連携インタフェース（３０）にサービスエージェント（２０）を送る（１１１７）。サービスエンティティ名が接続管理テーブル（４８）に登録されていなければ、次にサービスエンティティ管理テーブル（４７）にサービスエンティティ名が登録されているか否かをチェックする（１１１８）。サービスエンティティ管理テーブル（４７）にサービスエンティティ名が登録されていれば、そのサービスエンティティ（４）を起動する（１１１９）。起動されたサービスエンティティ（４）の連携インタフェース（３０）は、ルーティング

マネージャ（１０）に接続要求を行う。ルーティングマネージャ（１０）は、サービスエンティティ（４）の連携マネージャ（３０）からの接続要求を待ち（１１２０）、接続要求を受け取るとルーティングマネージャ（１０）と連携インタフェース（３０）間の入出力チャネル（４０３）を生成し（１１２１）、起動したサービスエンティティ名とその入出力チャネル（４０３）のチャネルポート番号を接続管理テーブル（４８）に登録し（１１２２）、生成した入出力チャネル（４０３）を通してサービスエージェント（２０）をサービスエンティティ（４）の連携インタフェース（３０）に送る（１１１７）。

【０１０２】接続管理テーブル（４８）、サービスエンティティ管理テーブル（４７）のいずれのテーブルにも登録されていなければ、図１５に移り、接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がルーティングマネージャである接続先があるか否かチェックする（１１２３）。接続先が登録されていれば、その中の最初に登録されている接続先名（４８２）を取りだし（１１２４）、その接続先名（４８２）が移動リスト一覧（２０３）に格納された移動リスト（１０１）の移動コンピュータ名（１０２）にあるか否かをチェックする（１１２５）。移動リスト（１０１）になければ、移動リスト（１０１）の移動コンピュータ名（１０２）に現在のコンピュータの名称を追加し（１１２６）、追加した移動リスト（１０１）をサービスエージェント（２０）の移動リスト一覧（２０３）に格納し（１１２７）、その接続先名（４８２）のチャネルポート番号フィールド（４８１）に格納されているチャネルポート番号（４８３）を取りだし（１１２８）、そのチャネルポート番号に接続されたチャネル（４０）を通してチャネルに接続されているコンピュータのルーティングマネージャ（１０）にサービスエージェント（２０）を送る（１１２９）。移動リスト（１０１）にあれば、接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がルーティングマネージャである次の接続先があるか否かチェックする（１１２３）。次の接続先が接続管理テーブル（４８）に登録されていれば、その接続先名（４８２）を取りだし（１１２４）、その接続先名（４８２）がサービスエージェント（２０）の移動リスト一覧（２０３）に格納されている移動リスト（１０１）の移動コンピュータ名（１０２）にあるか否かをチェックする（１１２５）。これを接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がルーティングマネージャである接続先がなくなるか、移動リスト（１０１）にない接続先名（４８２）が見つかるまで順次繰り返す。その結果、接続管理テーブル（４８）の接続先種別フィールド（４８４）の接続先種別名（４８５）がルーティングマネージャである接続先がなくなった場合、エラーとしてそのサービスエージェント（２０）の移動を終了する（１１３０）。

【０１０３】図１６に移り、サービスエージェント（２０）を受け取ったサービスエンティティ（４）の連携インタフェース（３０）は、受け取ったサービスエージェント（２０）に記述されているデータ出力手続き（２３）を実行しサービスエージェント（２０）からサービスエンティティ（４）に送るデータを取得し（１１３１）、制御手続き（２６）を実行して、そのサービスエンティティ（４）の持つアプリケーションプログラム本体（３）の実行を制御し（１１３２）、その実行結果データをサービスエージェント（２０）のデータ入力手続き（２２）を実行してサービスエージェント（２０）に格納し（１１３３）、サービスエージェント（２０）の移動手続き（２５）を実行して移動先名（２０４）の値を次の移動先に書き換え（１１３４）、移動先名（２０４）の値を書き換えたサービスエージェント（２０）を、連携インタフェース（３０）からルーティングマネージャ（１０）へ入出力チャネル（４０３）を通して送る（１１３５）。図１３に戻り、ルーティングマネージャ（１０）は、連携インタフェース（３０）から送られたサービスエージェント（２０）を入出力チャネル（４０３）を通して受け取る（１１００）と、チャネル（４０）を通してサービスエンティティ（２０）を受け取ったときと同様に、移動先名（２０４）の移動先コンピュータ名（９１２）をチェックし（１１０１）、サービスエージェント（２０）を次の移動先にチャネル（４０）を通して送る（１１０４）。このサービスエージェントの移動処理をサービスエージェント（２０）の移動先リスト（９１）の最後の要素になるまで、ルーティングマネージャ（１０）間で繰り返すことにより、サービスエージェント（２０）をサービスエンティティ（４）間で移動させ、移動させたサービスエージェント（２０）をサービスエンティティ（４）の連携インタフェース（３０）で処理することにより各アプリケーションプログラム本体（３）の実行を制御する。

【０１０４】図１７～図２０にサービスエンティティ（４）の連携インタフェース（３０）において、受け取ったサービスエージェント（２０）の持つデータや手続きを利用しながら、アプリケーションプログラム本体（３）の実行を制御するための処理フローを示す。

【０１０５】まず、図１７において、連携インタフェース（３０）は、入出力チャネル（４０３）を通じてルーティングマネージャ（１０）からサービスエージェント（２０）を受け取る（１２００）と、受け取ったサービスエージェント（２０）の出力データ選択手続き（２３１）を実行して入力済みデータ項目（２７０）の中からアプリケーションプログラム本体（３）に渡すデータを格納しているデータ項目である出力データ項目（２７１）を決定する（１２０１）。

【０１０６】出力データ項目（２７１）が決定されると、サービスエージェント（２０）の出力データ変換手続き（２３２）がプログラムであるか関数名であるかをチェックする（１２０２）。出力データ変換手続き（２３２）がプログラムであれば、その出力データ変換手続き（２３２）がアプリケーションプログラム本体（３）へのデータ出力関数か画面出力関数かをチェックし（１２０３）、データ出力関数であれば、その関数を実行して、出力データ項目（２７１）に格納されているデータからアプリケーションプログラム本体（３）に渡すデータを生成する（１２０４）。画面出力関数であれば、その関数を実行して、出力データ項目（２７１）に格納されているデータから画面に出力するデータを作成する（１２０５）。生成されたデータには、どちらのデータであるかを示すタグが付加される（１２０６）。出力データ変換手続き（２３２）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２０７）、取得した関数がアプリケーションプログラム本体（３）へのデータ出力関数か画面出力関数かをチェックする（１２０３）。

【０１０７】タグが付加されたデータは、タグからデータを画面に出力するか、アプリケーションプログラム本体（３）に渡すかが判定され（１２０８）、画面に出力するデータの場合にはサービスエージェント（２０）の出力データ手続き（２３４）がプログラムであるか関数名であるかがチェックされる（１２０９）。出力データ手続き（２３４）がプログラムであれば、その出力データ手続き（２３４）を実行し、画面出力データを画面に出力する（１２１０）。出力データ手続き（２３４）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２１１）、取得した関数を実行し、画面出力データを画面に出力する（１２１０）。アプリケーションプログラム本体（３）に渡すデータの場合、そのデータのタグを削除する（１２１２）。

【０１０８】次に図１８に移り、サービスエージェント（２０）の制御手続き（２６）がプログラムであるか関数名であるかがチェックされる（１２１３）。制御手続き（２６）がプログラムであれば、その制御手続き（２６）を実行することにより、アプリケーションプログラム本体（３）の実行を制御する（１２１４）。制御手続き（２６）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２１５）、取得した関数を実行することにより、アプリケーションプログラム本体（３）の実行を制御する（１２１４）。

【０１０９】制御手続き（２６）によりアプリケーションプログラム本体（３）の実行制御が終了すると、サービスエージェント（２０）のデータ取得手続き（２２３）がプログラムであるか関数名であるかがチェックされる（１２１６）。データ取得手続き（２２３）がプログラムであれば、そのデータ取得手続き（２２３）を実行し、アプリケーションプログラム本体（３）の実行結果データを取得する（１２１７）。データ取得手続き（２２３）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２１８）、取得した関数を実行し、アプリケーションプログラム本体（３）の実行結果データを取得する（１２１７）。

【０１１０】画面が出力されている場合には、図２０に移り、サービスエージェント（２０）の制御手続き（２６）がプログラムであるか関数名であるかがチェックされる（１２１９）。制御手続き（２６）がプログラムであれば、その制御手続き（２６）を実行することにより、画面からの入出力を制御する（１２２０）。制御手続き（２６）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２２１）、取得した関数を実行することにより、画面からの入出力を制御する（１２２０）。

【０１１１】画面が出力されている場合には、データ取得手続き（２２３）がプログラムであるか関数名であるかがチェックされる（１２２２）。データ取得手続き（２２３）がプログラムであれば、そのデータ取得手続き（２２３）を実行し、画面からの入力データを取得する（１２２３）。すなわち、画面を介してユーザの入力する指示やデータを取り込む。データ取得手続き（２２３）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２２４）、取得した関数を実行し画面からの入力データを取得する（１２２３）。

【０１１２】次に図１８に戻り、サービスエージェント（２０）の入力データ選択手続き（２２１）がプログラムであるか関数名であるかがチェックされる（１２２５）。入力データ選択手続き（２２１）がプログラムであれば、その入力データ選択手続き（２２１）を実行し、サービスエージェント（２０）の全データ項目（２１）の中からアプリケーションプログラム本体（３）の処理結果データを格納するデータ項目である入力データ項目（２８１）を選択する（１２２６）。入力データ選択手続き（２２１）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２２７）、取得した関数を実行し、サービスエージェント（２０）の全データ項目（２１）の中からアプリケーションプログラム本体（３）の処理結果データを格納するデータ項目である入力データ項目（２８１）を選択する（１２２６）。

【０１１３】次に図１９に移り、サービスエージェント（２０）の入力データ処理手続き（２４）がプログラムであるか関数名であるかがチェックされる（１２２８）。入力データ処理手続き（２４）がプログラムであ

れば、その入力データ処理手続き（２４）を実行し、受け取ったデータ及び全項目データ（２１）に格納されているデータを処理する（１２２９）。入力データ処理手続き（２４）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２３０）、取得した関数を実行し、受け取ったデータ及び全項目データ（２１）に格納されているデータを処理する（１２２９）。

【０１１４】次にサービスエージェント（２０）の入力データ変換手続き（２２４）がプログラムであるか関数名であるかがチェックされる（１２３１）。入力データ変換手続き（２２４）がプログラムであれば、その入力データ変換手続き（２２４）を実行し、受け取ったデータをそれに該当する入力データ項目（２８１）のデータ形式に変換し（１２３２）、変換したデータを入力データ項目（２８１）のそれぞれのデータ項目に格納する（１２３４）。入力データ変換手続き（２２４）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２３３）、取得した関数を実行し受け取ったデータをそれに該当する入力データ項目（２８１）のデータ形式に変換し（１２３２）、変換したデータを入力データ項目（２８１）のそれぞれのデータ項目に格納する（１２３４）。

【０１１５】入力データ項目（２８１）にデータが格納されると、サービスエージェント（２０）の移動手続き（２５）がプログラムであるか関数名であるかがチェックされる（１２３５）。移動手続き（２５）がプログラムであれば、その移動手続き（２５）を実行し、移動先名（２０４）の値を次の移動先に書き換え（１２３６）、移動先名（２０４）の値を書き換えたサービスエージェント（２０）を、連携インタフェース（３０）からルーティングマネージャ（１０）へ入出力チャネル（４０３）を通して送る（１２３７）。移動手続き（２５）が関数名であれば、その関数名のプログラムを関数制御プログラム（３７）を通して関数テーブルから取得し（１２３８）、取得した関数を実行しを実行し、移動先名（２０４）の値を次の移動先に書き換え（１２３６）、移動先名（２０４）の値を書き換えたサービスエージェント（２０）を、連携インタフェース（３０）からルーティングマネージャ（１０）へ入出力チャネル（４０３）を通して送る（１２３７）。

【０１１６】次に、本発明の具体的な適用例として、製造ライン管理システムへ適用した例について説明する。製造ライン管理システムでは、製造・検査機器、それら機器の制御用アプリケーションプログラム、生産管理に必要なアプリケーションプログラムを製造物ごとに決められた順序で連携させる必要がある。

【０１１７】図２２に、本発明による製造ライン管理システムの一適用例のシステム構成を示す。本適用例では、製造ライン管理システムが各製造工程Ａ、Ｂを管理するコンピュータＣ２（１３４２）、Ｃ３（１３４３）、各製造工程Ａ、Ｂに接続された製造機器ＭＡ１（１３３２）、ＭＢ１（１３３３）と、これらの製造機器ＭＡ１（１３３２）、ＭＢ１（１３３３）を制御するアプリケーションプログラムＡＰ２（１３０２）、ＡＰ３（１３０３）、コンピュータＣ１（１３４１）上で製造指示書を作成するアプリケーションプログラムＡＰ１（１３０１）、コンピュータを接続するネットワーク（０）から構成されるものとし、コンピュータＣ１（１３４１）上のアプリケーションプログラムＡＰ１（１３０１）で作成された製造指示書は、各製造工程に接続された製造機器をそれらの制御アプリケーションプログラムを通して連携制御することにより、製造を進めていく場合を例に取り説明する。

【０１１８】本実施例による製造ライン管理システムでは、これらの各種アプリケーションプログラムＡＰ１（１３０１）、ＡＰ２（１３０２）、ＡＰ３（１３０３）をサービスエンティティＳＥ１（１３２１）、ＳＥ２（１３２２）、ＳＥ３（１３２３）（各サービスエンティティの名称はそれぞれＳＥ１、ＳＥ２、ＳＥ３とする）として実現する。製造物に添付される製造指示書はサービスエージェントＳＡ１（１４００）として具現する。製造指示書を具現したサービスエージェントＳＡ１（１４００）は、サービスエンティティＳＥ１（１３２１）を用いて利用者により作成され、まず製造工程Ａの製造機器ＭＡ１（１３３３）を制御する制御用アプリケーションプログラムＡＰ２（１３０２）を持つサービスエンティティＳＥ２（１３２２）を通してアプリケーションプログラムＡＰ２（１３０２）に製造指示を伝え、次に製造工程Ｂの製造機器ＭＢ１（１３３３）を制御する制御用アプリケーションプログラムＡＰ３（１３０３）を持つサービスエンティティＳＥ３（１３２３）を通してアプリケーションプログラムＡＰ３（１３０３）に製造指示を伝え、その後サービスエージェントＳＡ１（１４００）を作成したサービスエンティティＳＥ１（１３２１）に戻るものとする。

【０１１９】図２３に利用者がサービスエンティティＳＥ１（１３２１）を用いてコンピュータＣ１（１３４１）上で作成した時点の製造指示書を表すサービスエージェントの一例であるサービスエージェントＳＡ１（１４００）の構造を示す。サービスエージェントＳＡ１（１４００）は、識別子ＩＤ１（１４０１）、移動先リスト一覧（（Ｃ２　ＳＥ２）（Ｃ３　ＳＥ３）（Ｃ１　ＳＥ１））（１４０２）、移動リスト一覧ｎｉｌ（１４０３）、移動先名（Ｃ２　ＳＥ２）（１４０４）に加え、全データ項目として、製造機器ＭＡ１の設定パラメータである項目であるＡ工程製造速度項目（１４０５）とその値１０、製造機器ＭＡ１から取得するデータの項目であるＡ工程所要時間項目（１４０６）、製造機器Ｍ

Ｂ１の設定パラメータである項目であるＢ工程製造速度項目（１４０７）とその値１５、製造機器ＭＢ１から取得するデータの項目であるＢ工程所要時間項目（１４０８）を持つ。さらにデータ出力手続きとして、出力データ選択手続き（１４２１）、出力データ変換手続き（１４２２）、パラメタ設定手続き（１４２３）を、データ入力手続きとして、データ取得手続き（１４１１）、入力データ選択手続き（１４１２）、入力データ格納手続き（１４１３）を、さらに制御手続き（１４３０）を持つ。

【０１２０】サービスエージェントＳＥ１（１３２１）を用いて生成されたサービスエージェントＳＡ１（１４００）は、チャネル（４０）を通してコンピュータＣ１（１３４１）のルーティングマネージャＲＭ１（１３１１）に送られる。ルーティングマネージャＲＭ１（１３１１）は、サービスエージェントＳＡ１（１４００）の移動先名の値（Ｃ２　ＳＥ２）（１４０４）をチェックし、移動先名に指定されたコンピュータＣ２（１３４２）との間に接続されたチャネル（４０）を検索し、その結果得られたチャネル（４０）を通してコンピュータＣ２（１３４２）にサービスエージェントＳＡ１（１４００）を送る。

【０１２１】コンピュータＣ２（１３４２）上のルーティングマネージャＲＭ２（１３１２）は、送られてきたサービスエージェントＳＡ１（１４００）を受け取ると、その移動先名の値（Ｃ２　ＳＥ２）（１４０４）をチェックし、移動先名に指定されたサービスエンティティＳＥ２（１３２２）との間のに接続されたチャネル（４０）を検索し、その結果得られたチャネル（４０）を通してサービスエンティティＳＥ２（１３２２）にサービスエージェントＳＡ１（１４００）を送る。サービスエンティティＳＥ２（１３２２）は、チャネル（４０）を通してサービスエージェントＳＡ１（１４００）を受け取ると、サービスエンティティＳＥ２（１３２２）の連携インタフェース（３０）がサービスエージェントＳＡ１（１４００）の持つ出力データ選択手続き（１４２１）を取得し、その手続き（１４２１）を実行し、サービスエンティティＳＥ２（１３２２）の持つアプリケーションプログラムＡＰ２（１３０２）に渡すべきデータ項目であるＡ工程製造速度項目（１４０５）を決定する。次に、サービスエージェントＳＡ１（１４００）の持つ出力データ変換手続き（１４２２）を取得し、その手続き（１４２２）を実行し、選択された項目であるＡ工程製造速度項目（１４０５）の値１０を取り出し、そのデータがアプリケーションプログラムに渡すデータであることを示すタグを付加した後、サービスエージェントＳＡ１（１４００）の持つパラメタ設定手続き（１４２３）を取得し、その手続き（１４２３）を実行し、アプリケーションプログラムＡＰ２（１３０２）に指定された値１０を設定する。次にサービスエージェントＳＡ１（１４００）の持つ制御手続き（１４３０）を取得し、その手続き（１４３０）を実行することによりアプリケーションプログラムＡＰ２（１３０２）の処理を行なう。

【０１２２】アプリケーションプログラムＡＰ２（１３０２）の処理が終了すると、サービスエージェントＳＡ１（１４００）の持つデータ取得手続き（１４１１）を取得し、その手続き（１４１１）を実行し、アプリケーションプログラムの実行結果を取得する。次にサービスエージェントＳＡ１（１４００）の持つ入力データ選択手続き（１４１２）を取得し、その手続き（１４１２）を実行し、アプリケーションプログラムの実行結果を格納するデータ項目であるＡ工程所要時間（１４０６）を選択し、次にサービスエージェントＳＡ１（１４００）の持つ入力データ格納手続き（１４１３）を取得し、その手続き（１４１３）を実行し、アプリケーションプログラムＡＰ２（１３０２）の実行結果をＡ工程所要時間項目（１４０６）に格納する。

【０１２３】次に移動先リスト一覧（１４０２）の２番目の要素である（Ｃ３　ＳＥ３）を取り出し、それを移動先名（１４０４）に格納し、チャネル（４０）を通してコンピュータＣ２（１３４２）上のルーティングマネージャＲＭ２（１３１２）に送る。

【０１２４】ルーティングマネージャＲＭ２（１３１２）は、サービスエンティティＳＥ２（１３２２）からチャネル（４０）を通して送られてきたサービスエージェントＳＡ１（１４００）を受け取ると、その移動先名（Ｃ３　ＳＥ３）（１４０４）をチェックし、移動先名（Ｃ３　ＳＥ３）（１４０４）に指定されたコンピュータＣ３（１３４３）に接続されたチャネル（４０）を検索し、その結果得られたチャネル（４０）を通してサービスエージェントＳＡ１（１４００）を送る。

【０１２５】コンピュータＣ３（１３４３）上のルーティングマネージャＲＭ３（１３１３）は、送られてきたサービスエージェントＳＡ１（１４００）を受け取ると、その移動先名の値（Ｃ３　ＳＥ３）（１４０４）をチェックし、移動先名（１４０４）に指定されたサービスエンティティＳＥ３（１３２３）との間に接続されたチャネル（４０）を検索し、その結果得られたチャネル（４０）を通してサービスエンティティＳＥ３（１３２３）にサービスエージェントＳＡ１（１４００）を送る。

【０１２６】サービスエンティティＳＥ３（１３２３）は、チャネル（４０）を通してサービスエージェントＳＡ１（１４００）を受け取ると、サービスエンティティＳＥ３（１３２３）の連携インタフェース（３０）がサービスエージェントＳＡ１（１４００）の持つ出力データ選択手続き（１４２１）を取得し、その手続き（１４２１）を実行し、サービスエンティティＳＥ３（１３２３）の持つアプリケーションプログラムＡＰ３（１３０

3）に渡すべきデータ項目であるＢ工程製造速度項目（１４０７）を決定する。次に、サービスエージェントＳＡ１（１４００）の持つ出力データ変換手続き（１４２２）を取得し、その手続き（１４２２）を実行し、選択された項目であるＢ工程製造速度（１４０７）の値１５を取り出し、そのデータがアプリケーションプログラムに渡すデータであることを示すタグを付加した後、サービスエージェントＳＡ１（１４００）の持つパラメタ設定手続き（１４２３）を取得し、その手続き（１４２３）を実行し、アプリケーションプログラムＡＰ３（１３０３）に指定された値１５を設定する。次にサービスエージェントＳＡ１（１４００）の持つ制御手続き（１４３０）を取得し、その手続き（１４３０）を実行することによりアプリケーションプログラムＡＰ３（１３０３）の処理を行なう。

【０１２７】アプリケーションプログラムＡＰ３（１３０３）の処理が終了すると、サービスエージェントＳＡ１（１４００）の持つデータ取得手続き（１４１１）を取得し、その手続き（１４１１）を実行し、アプリケーションプログラムの実行結果を取得する。次にサービスエージェントＳＡ１（１４００）の持つ入力データ選択手続き（１４１２）を取得し、その手続き（１４１２）を実行し、アプリケーションプログラムの実行結果を格納するデータ項目であるＢ工程所要時間項目（１４０８）を選択し、次にサービスエージェントＳＡ１（１４００）の持つ入力データ格納手続き（１４１３）を取得し、その手続き（１４１３）を実行し、アプリケーションプログラムＡＰ３（１３０３）の実行結果をＢ工程所要時間項目（１４０８）に格納する。

【０１２８】次に移動先リスト一覧（１４０２）の３番目の要素である（Ｃ１　ＳＥ１）を取り出し、それを移動先名（１４０４）に格納し、チャネル（４０）を通してコンピュータＣ３（１３４３）上のルーティングマネージャＲＭ３（１３１３）に送る。

【０１２９】ルーティングマネージャＲＭ３（１３１３）は、サービスエンティティＳＥ３（１３２３）からチャネル（４０）を通して送られてきたサービスエージェントＳＡ１（１４００）を受け取ると、その移動先名（Ｃ１　ＳＥ１）（１４０４）をチェックし、移動先名（Ｃ１　ＳＥ１）（１４０４）に指定されたコンピュータＣ１（１３４１）に接続されたチャネル（４０）を検索するが、コンピュータＣ３（１３４３）上のルーティングマネージャＲＭ３（１３１３）と接続されているコンピュータの中にはコンピュータ名Ｃ１（１３４１）がないため、接続されているコンピュータＣ２（１３４２）のルーティングマネージャＲＭ２（１３１２）にサービスエージェントＳＡ１（１４００）を送る。

【０１３０】コンピュータＣ２（１３４２）上のルーティングマネージャＲＭ２（１３１２）は、送られてきたサービスエージェントＳＡ１（１４００）を受け取ると、その移動先名の値（Ｃ１　ＳＥ１）（１４０４）をチェックし、移動先名（１４０４）に指定されたコンピュータ名Ｃ１がコンピュータ名Ｃ２と異なるため、サービスエージェントＳＡ１（１４００）の移動リスト（１４０３）にコンピュータ名Ｃ２を格納し、移動リスト（１４０３）の値を（（Ｃ２））とし、サービスエージェントＳＡ１（１４００）の移動先名（Ｃ１　ＳＥ１）（１４０４）に指定されたコンピュータ名Ｃ１と接続されたチャネル（４０）を検索し、その結果得られたチャネル（４０）にサービスエージェントＳＡ１（１４００）を送る。

【０１３１】ルーティングマネージャＲＭ１（１３１１）は、ルーティングマネージャＲＭ２（１３１２）からチャネル（４０）を通して送られてきたサービスエージェントＳＡ１（１４００）を受け取ると、その移動先名（Ｃ１　ＳＥ１）（１４０４）をチェックし、移動先名（Ｃ１　ＳＥ１）（１４０４）とコンピュータ名Ｃ１が同一であることをチェックした後、移動先名（Ｃ１　ＳＥ１）（１４０４）に指定されたサービスエンティティＳＥ１（１３２１）との間に接続されたチャネル（４０）を検索し、その結果得られたチャネル（４０）を通してサービスエンティティＳＥ１（１３２１）にサービスエージェントＳＡ１（１４００）を送る。

【０１３２】このようにしてサービスエージェントＳＡ１（１４００）を用いることにより、製造工程Ａ、Ｂを管理するコンピュータＣ２（１３４２）、Ｃ３（１３４３）に接続された製造機器ＭＡ１（１３３２）、ＭＢ１（１３３３）を制御するアプリケーションプログラムＡＰ２（１３０２）、ＡＰ３（１３０３）を連携して制御することが可能となる。また、製造工程が異なる場合でも、製造指示書を作成する際に、その移動先リスト一覧（１４０２）に指定する移動先の一覧を変更するだけで制御する製造装置やその順序を変更することができ、容易に連携の方法を変更することができる。

【０１３３】

【発明の効果】以上のように、本発明では、さまざまなアプリケーションプログラムを柔軟に連携処理するために、アプリケーションプログラム本体に連携処理のための連携インタフェースを付与し、連携のために必要な連携処理手続きと連携する際に必要なデータ、及びデータの格納項目、個々のアプリケーションプログラム本体の実行の制御手続きを作業指示書に記述し、その作業指示書をコンピュータ間で移動させることにより、柔軟なアプリケーションプログラム間の連携処理が可能となる。

【０１３４】これにより、連携処理の対象とするアプリケーションプログラムの実行を停止することなく、アプリケーションプログラム間の連携処理手続きを作業指示書により変更することが可能となる。また、利用者は連携処理をしたいアプリケーションプログラムが分散処理システム上のどのコンピュータで動作しているかを意識

することなく、利用したいアプリケーションプログラム本体（サービスエンティティ名）を指定することで、そのアプリケーションプログラムを利用することができる。

【０１３５】また、複数の作業者に依頼し、各作業者にアプリケーションプログラムを操作して行ってもらっていた一連の作業を自動化することができる。

【図面の簡単な説明】

【図１】本発明によるアプリケーションプログラム間連携処理を行う分散処理システムの実施例の構成を示す構成図である。

【図２】図１に示したコンピュータの構成を示す構成図である。

【図３】実施例におけるサービスエージェントの構造を示す説明図である。

【図４】実施例ににおけるルーティングマネージャ間の通信路の接続状態の説明図である。

【図５】図４の接続状態に対して新たなルーティングマネージャが接続された状態を示す説明図である。

【図６】実施例におけるルーティングマネージャとサービスエンティティの接続状態の説明図である。

【図７】実施例におけるルーティングマネージャのプログラム構造を示すブロック図である。

【図８】図７に示したテーブル類の構造の説明図である。

【図９】実施例における連携インタフェース３０のプログラム構造を示すブロック図である。示した

【図１０】図９に示した関数テーブル３７１の構造の説明図である。

【図１１】図３に示した移動先リスト一覧２０２の移動先リスト９１の記述形式および記述例の説明図である。

【図１２】図３に示した移動リスト一覧２０３の移動リスト１０１の記述方法および記述例の説明図である。

【図１３】実施例におけるサービスエージェントの移動処理手続きを示すフローチャート（その１）を示す。

【図１４】実施例におけるサービスエージェントの移動処理手続きを示すフローチャート（その２）を示す。

【図１５】実施例におけるサービスエージェントの移動処理手続きを示すフローチャート（その３）を示す。

【図１６】実施例におけるサービスエージェントの移動処理手続きを示すフローチャート（その４）を示す。

【図１７】実施例における連携インタフェース３０の処理を示すフローチャート（その１）である。

【図１８】実施例における連携インタフェース３０の処理を示すフローチャート（その２）である。

【図１９】実施例における連携インタフェース３０の処理を示すフローチャート（その３）である。

【図２０】実施例における連携インタフェース３０の処理を示すフローチャート（その４）である。

【図２１】実施例におけるデータ項目の組み合わせによる移動先決定方法の例の説明図である。

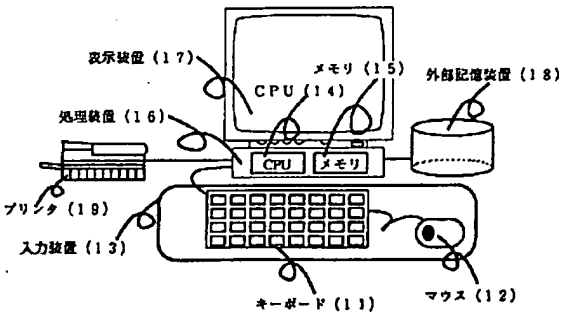
【図２２】本発明を適用した製造ライン管理システムの構成例を示す構成図である。

【図２３】図２２のシステムにおけるサービスエージェントの構造例を示す説明図である。

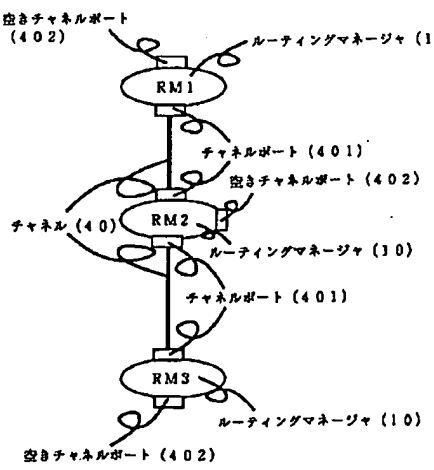
【符号の説明】

１…コンピュータ、３…アプリケーションプログラム本体、４…サービスエンティティ、１０…ルーティングマネージャ、２０…サービスエージェント、３０…連携インタフェース。

【図２】

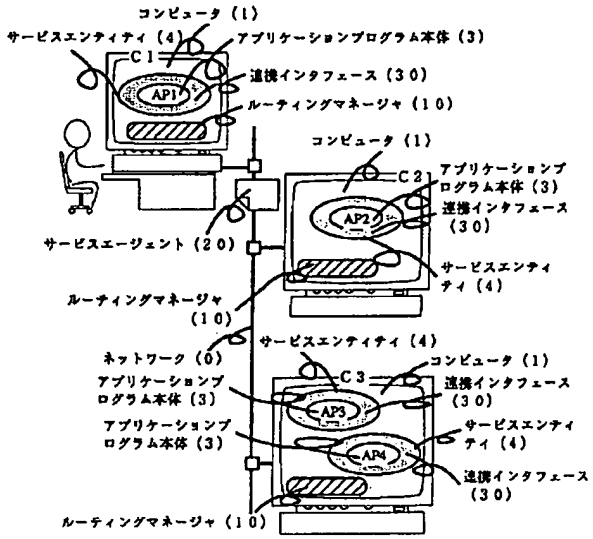


【図４】



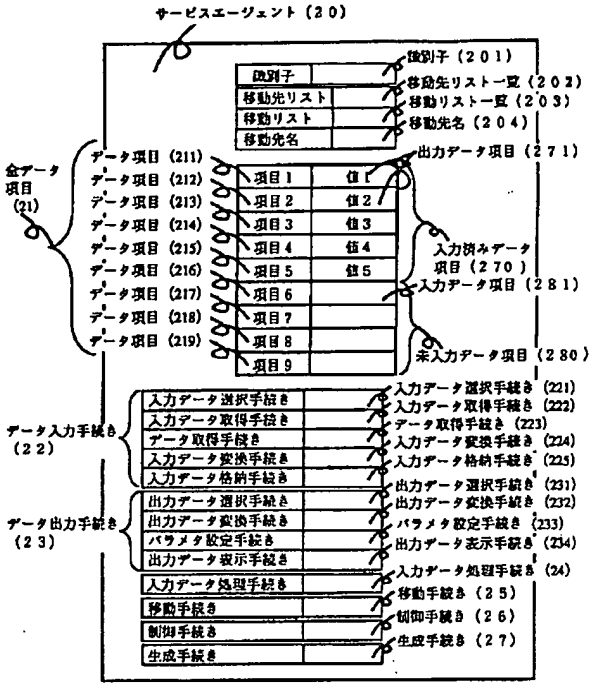
【図１】

図１　アプリケーションプログラム間連携処理システム構成



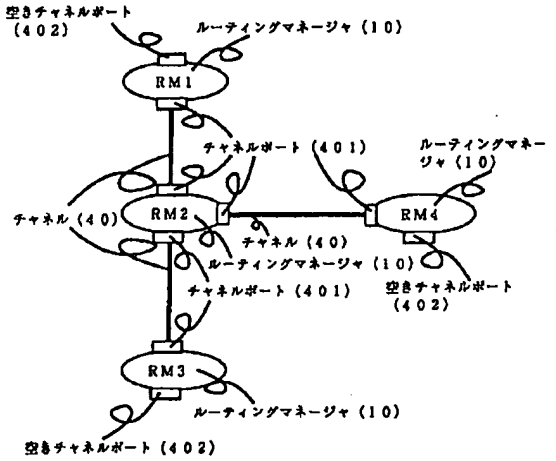
【図３】

図３　サービスエージェントの構造



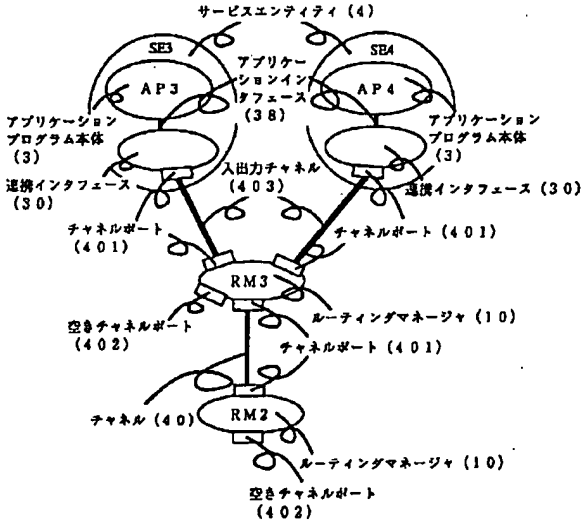
【図５】

図５　新たにルーティングマネージャが接続された状態



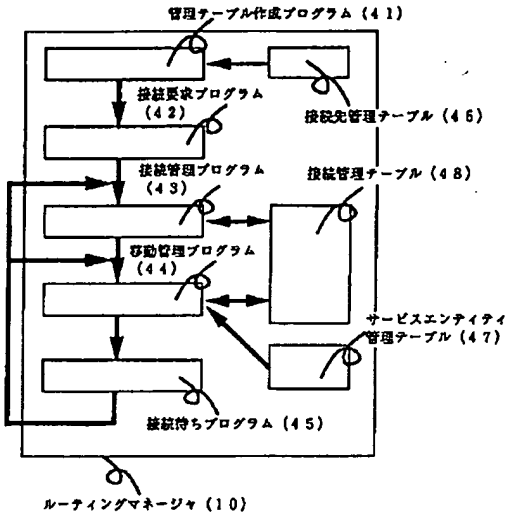
【図６】

図６　ルーティングマネージャとサービスエンティティの接続状態



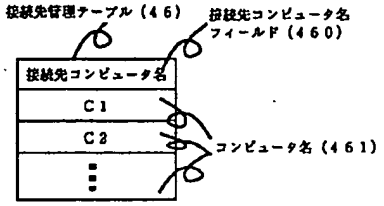
【図７】

図７　ルーティングマネージャのプログラム構造

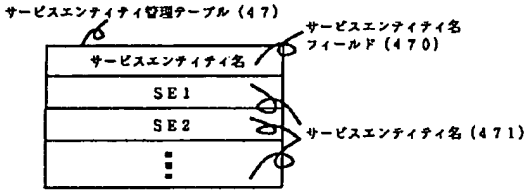


【図８】

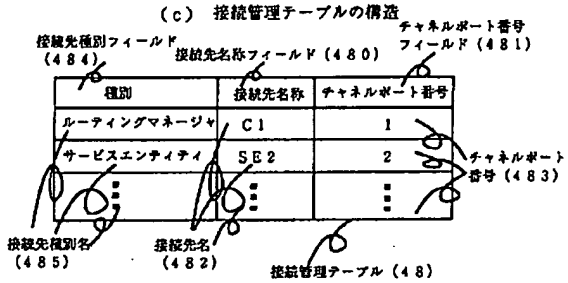
図８　（ａ）接続先管理テーブルの構造



（ｂ）サービスエンティティ管理テーブルの構造

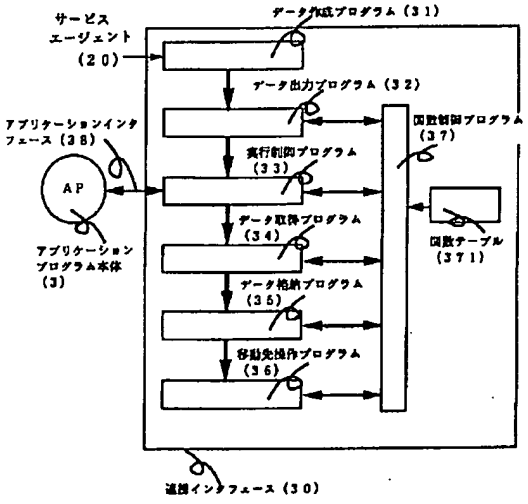


（ｃ）接続管理テーブルの構造



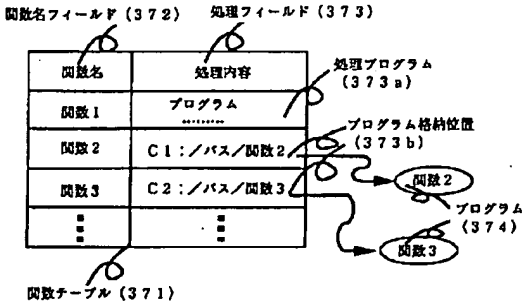
【図９】

図９　通信インタフェースのプログラム構造

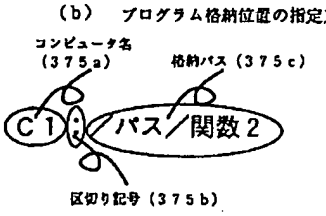


【図１０】

図１０　（ａ）関数テーブルの構造

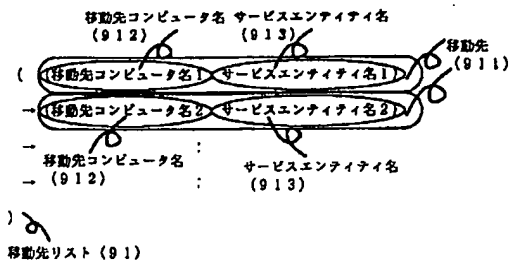


（ｂ）プログラム格納位置の指定方法

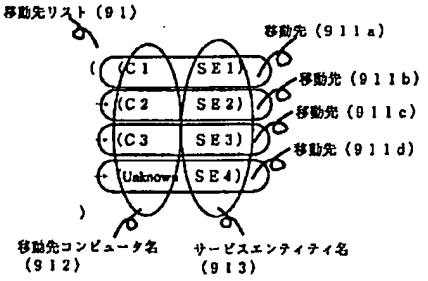


【図１１】

図１１　（ａ）移動先リストの記述形式

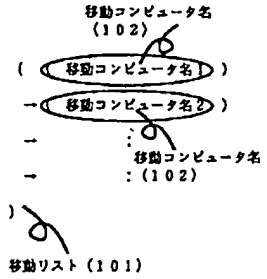


（ｂ）移動先リストの記述例

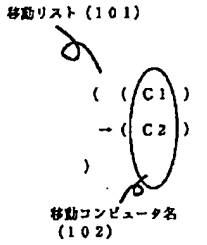


【図１２】

図１２　（ａ）移動リストの記述方法

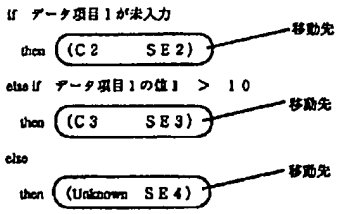


（ｂ）移動リストの記述例



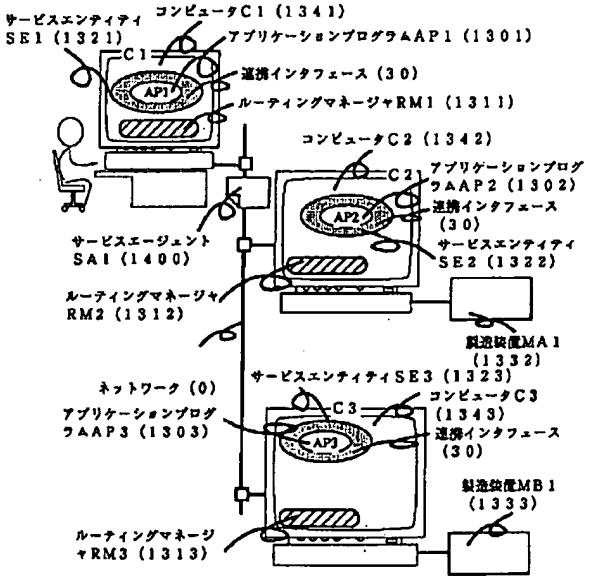
【図２１】

図２１　データ項目の組合せによる移動先決定方法の例



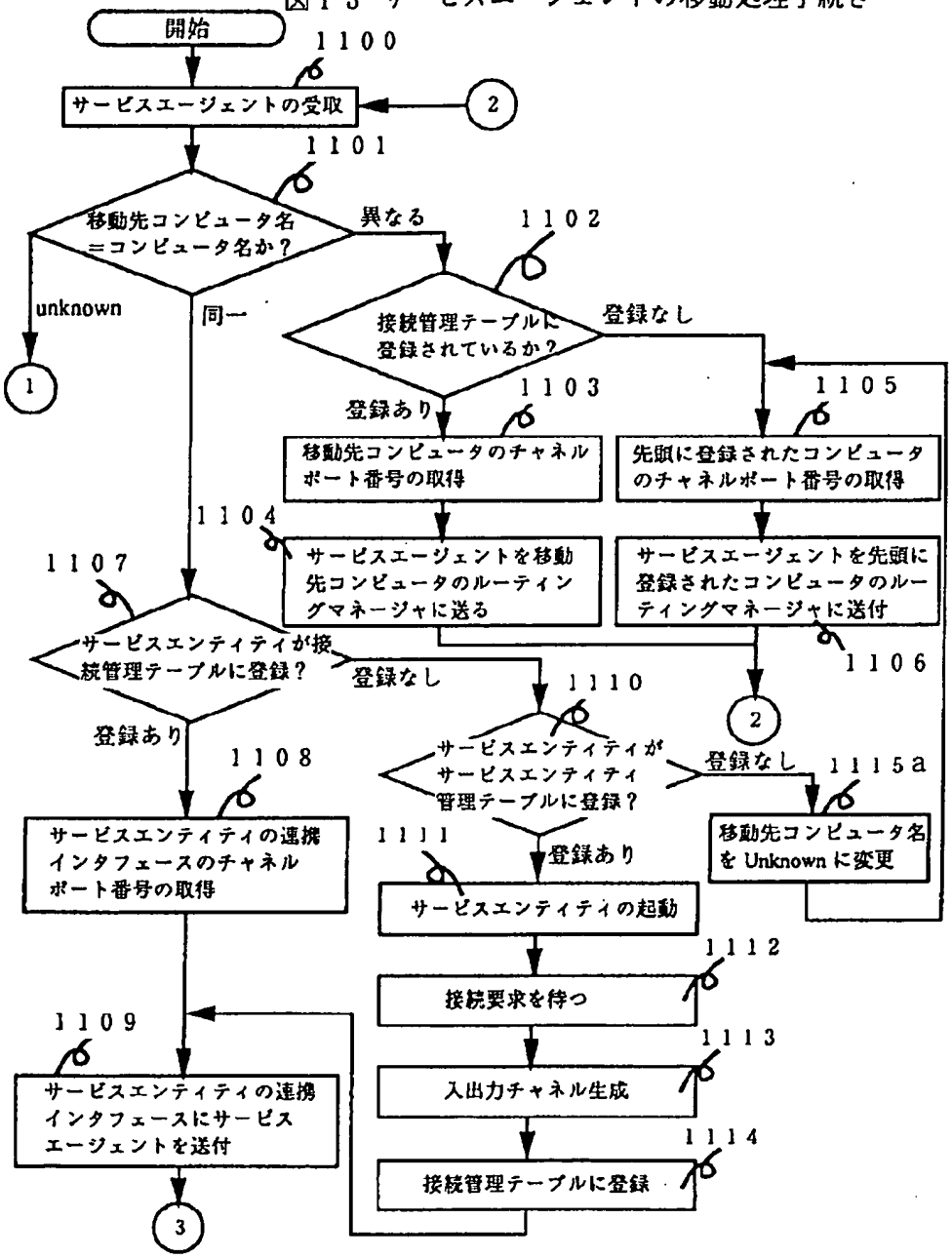
【図２２】

図２２　本発明による製造ライン管理システムの例



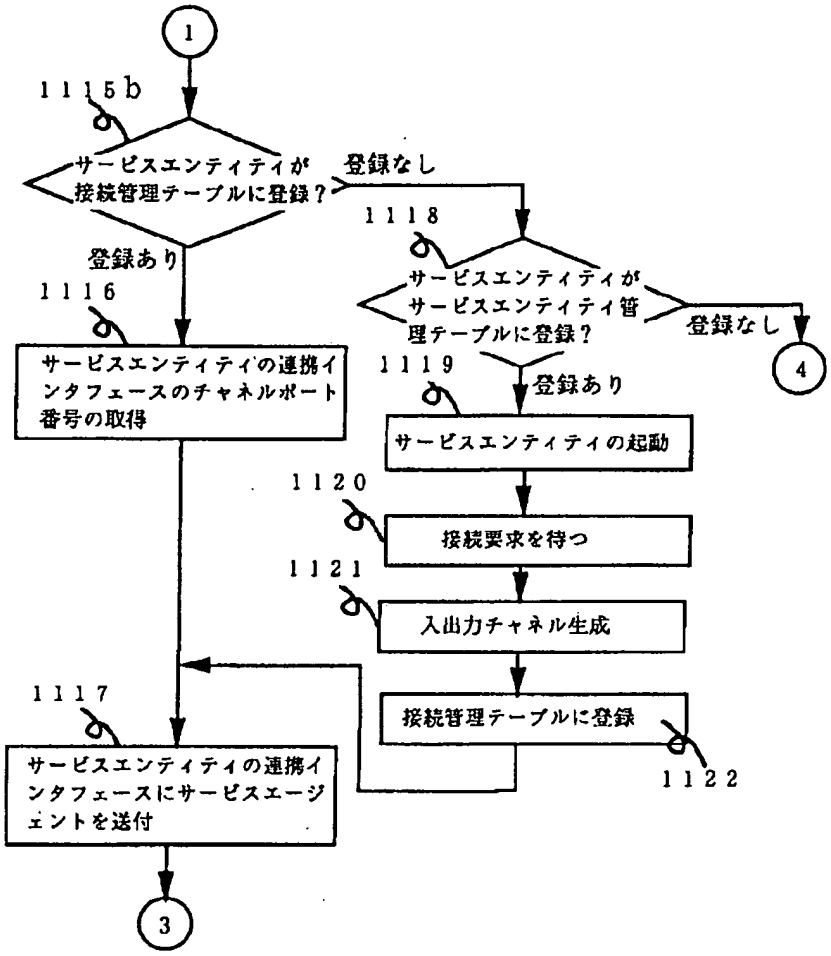
【図１３】

図１３ サービスエージェントの移動処理手続き



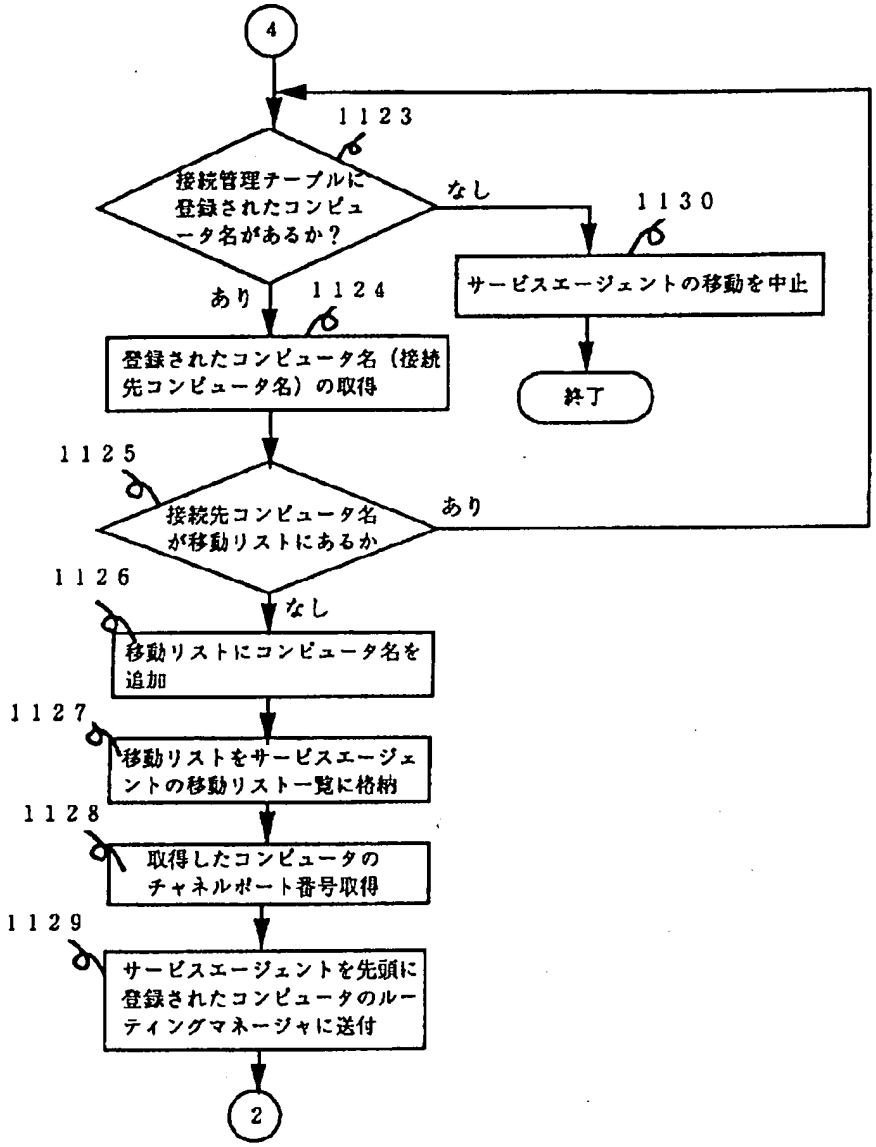
【図１４】

図１４　サービスエージェントの移動処理手続き



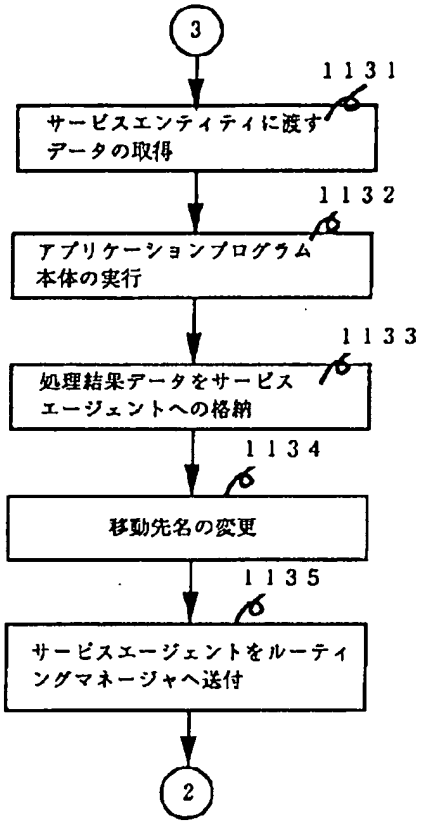
【図１５】

図１５　サービスエージェントの移動処理手続き



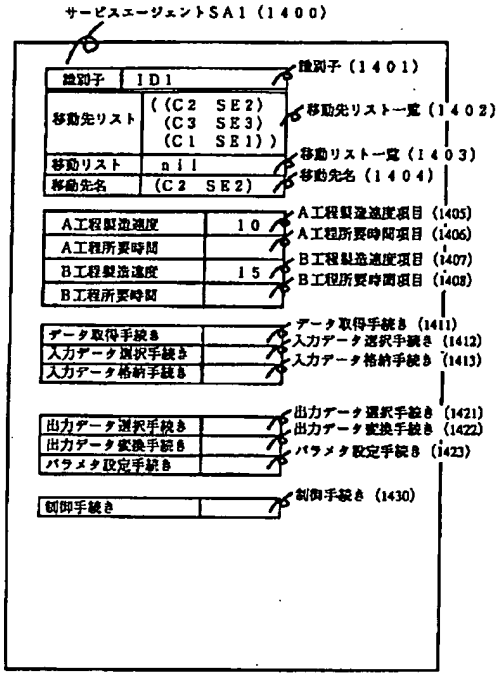
【図１６】

図１６　サービスエージェントの移動処理手続き



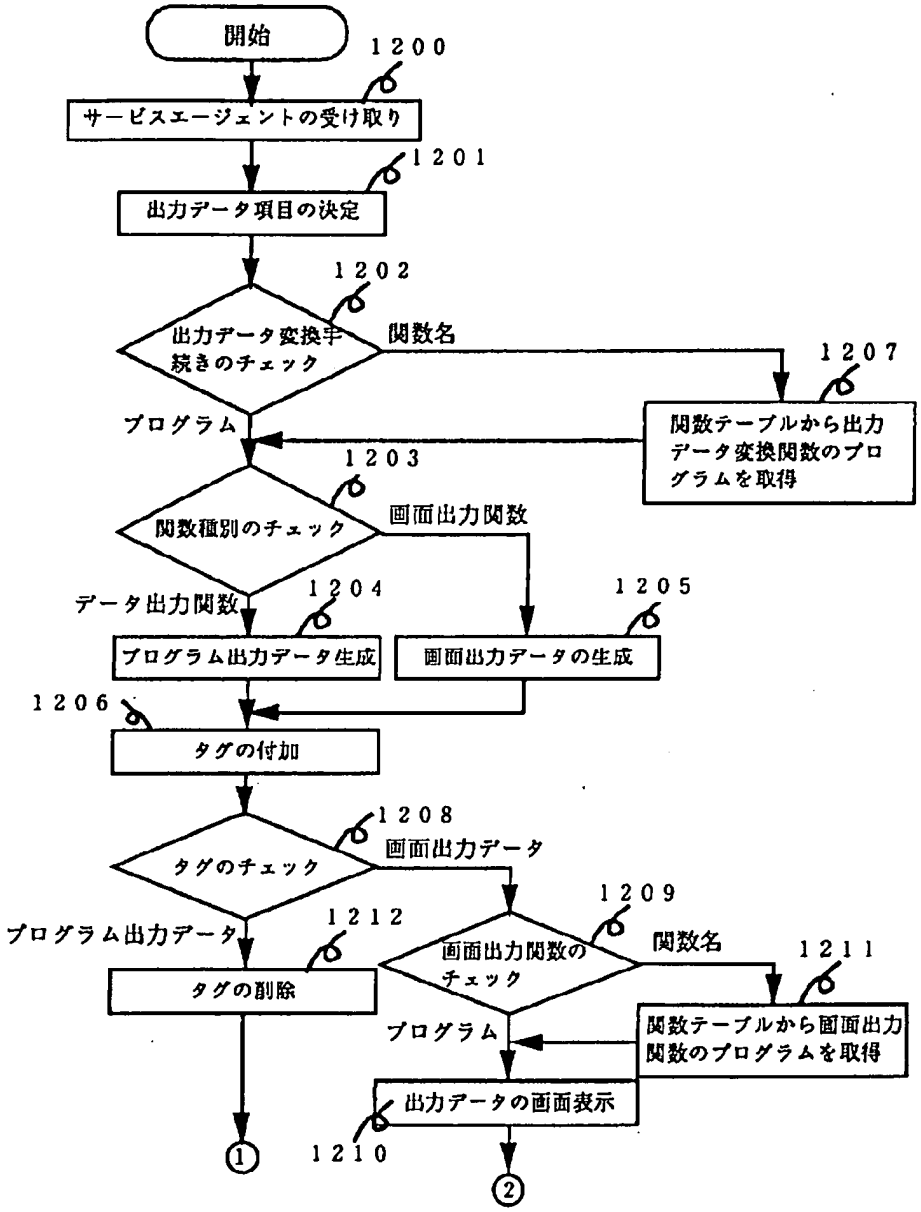
【図２３】

図２３　サービスエージェントＳＡ１の構造



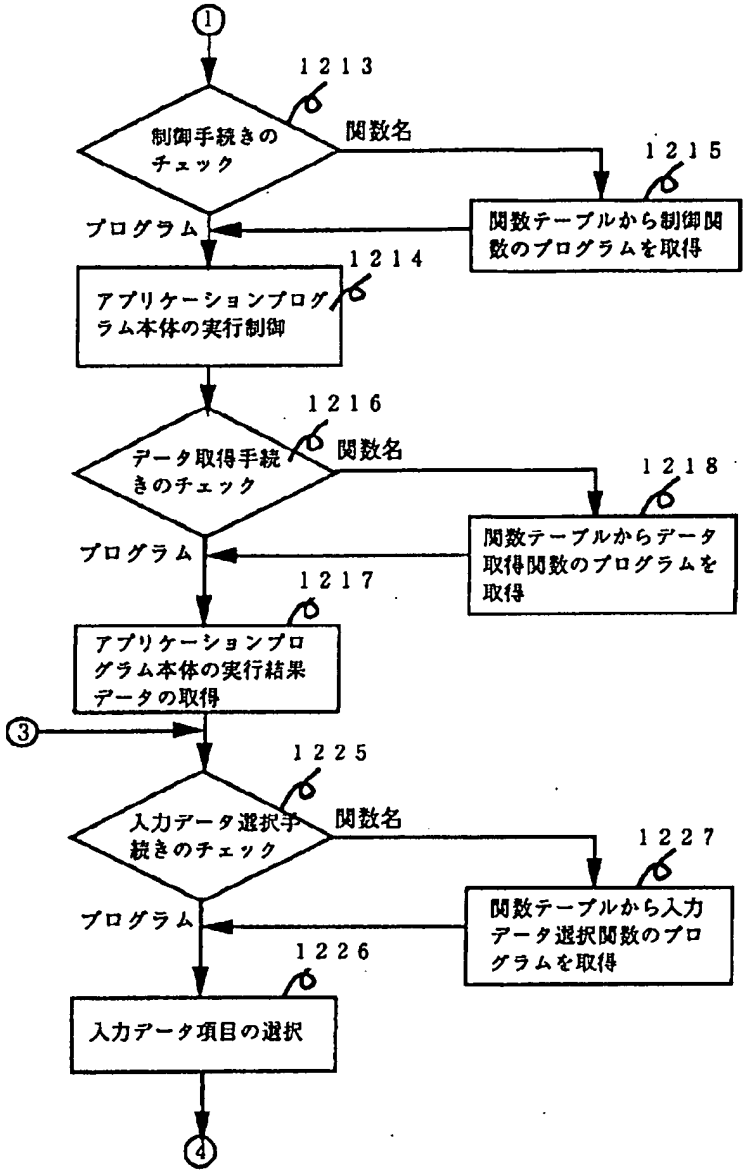
【図１７】

図１７　連携インタフェースの処理フロー



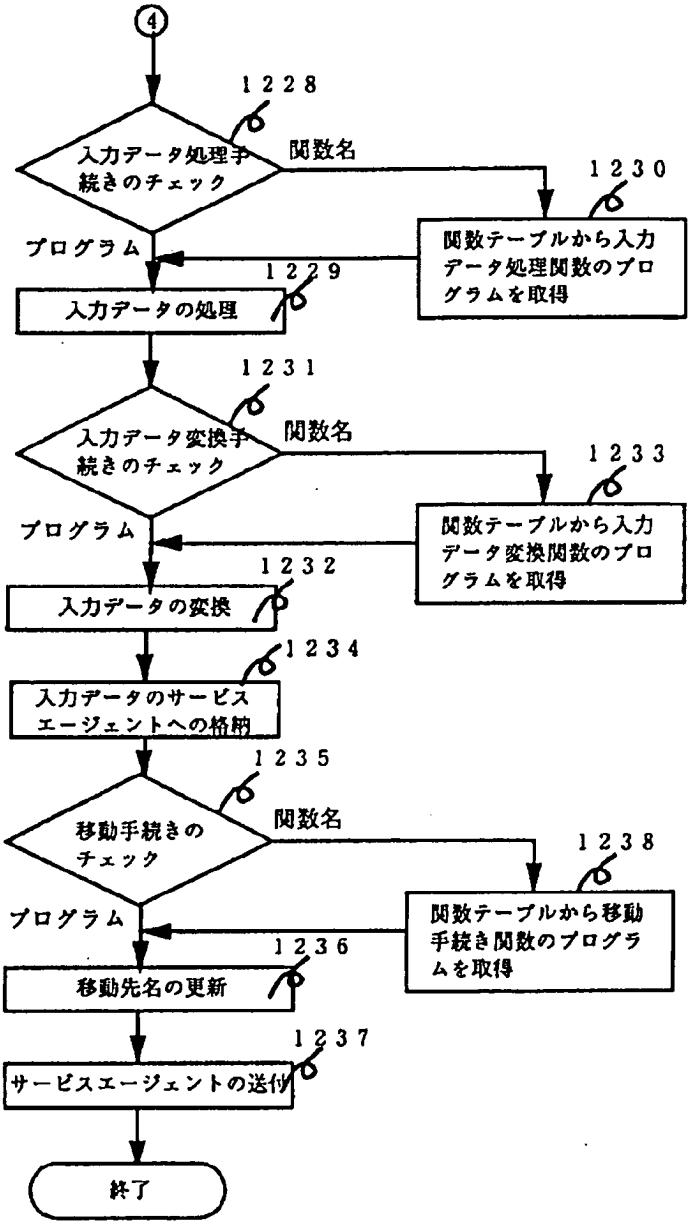
【図１８】

図１８　連携インタフェースの処理フロー



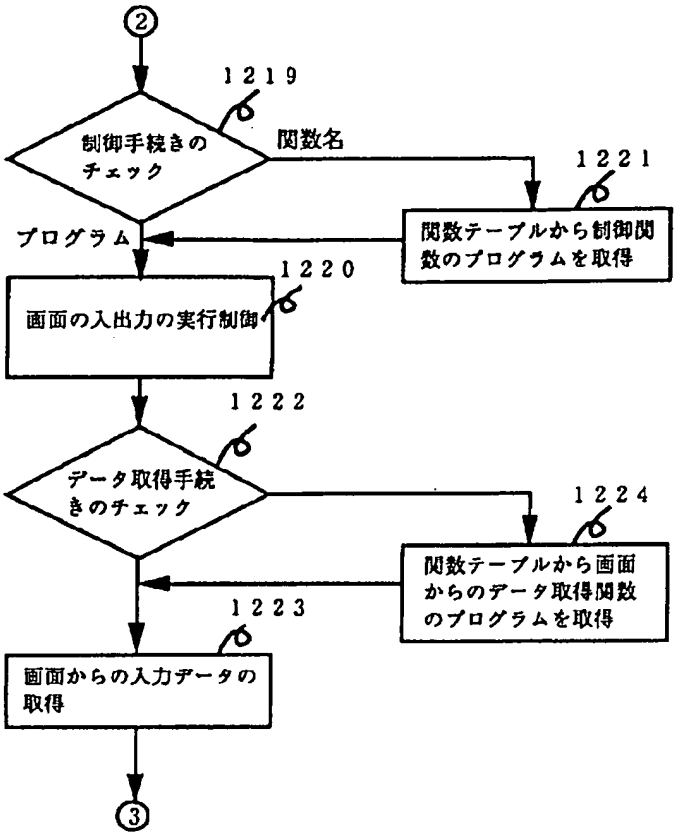
【図１９】

図１９　連携インタフェースの処理フロー



【図２０】

図２０　連携インタフェースの処理フロー



フロントページの続き

(72)発明者　高橋　勉
宮城県仙台市青葉区一番町二丁目４番１号
日立東北ソフトウェア株式会社内